Efficient Reliability Modelling & Analysis of Complex Systems with Application to Nuclear Power Plant Safety

Thesis submitted in fulfilment of the requirements of the University of Liverpool and the National Tsing Hua University, for the degrees of Doctor of Philosophy in Engineering and Nuclear Engineering, respectively

by

Hindolo George-Williams (B.Eng(Hons), Electrical/Electronic Engineering and M.Sc(Eng), Energy Generation)

May 2018

Dedication

I dedicate this piece of work to my wife, children, parents, and siblings.

Acknowledgement

I thank God the Almighty Father, for His unfailing love and divine favour during my research. I thank, also, the Liverpool School of Engineering and the College of Nuclear Science - National Tsing Hua University in Taiwan, for funding the research reported in this thesis. The series of trainings I received from the Engineering and Physical Sciences Research Council (EPSRC) Centre for Doctoral Training (CDT) on the Quantification and Management of Risk & Uncertainty in Complex Systems & Environments are invaluable. For this, I express my sincere gratitude to the management of the CDT.

I would be remiss to not acknowledge the unwavering support, guidance, and attention I received from my supervisors, Dr. Edoardo Patelli and Prof. Min Lee. They did an excellent job shaping the direction of my research and most importantly, putting up with my many questions and sometimes tenacious tendencies, when it came to my work. My relationship with Dr. Patelli, especially, has evolved over the years to transcend the usual student-supervisor relationship. This, on its own, went a long way to creating a relaxed and friendly research environment, one in which I expressed my opinions freely.

My research would not have been as interesting and smooth, without the love and support I received from my friends, parents, and siblings. They were always there for me, and for this, I thank them. To my wife Henrietta, words can not quite express how grateful I am for your understanding, love, and care. Even when it was this very research that kept us apart for years, you never complained, instead, you encouraged me to dedicate as much time to it as was required. You couldn't have been more supportive!

Finally, I thank Hector Diego Estrada-Lugo, Owen Kai-Combey, Ikenna Okaro, and Uchenna Oparaji, for proofreading this thesis.

Abstract

Nuclear power may be our best chance at a permanent solution to the world's energy challenges, owing to its sustainability and environmental friendliness. However, it also poses a great risk to life, property, and the economy, given the possibility of severe accidents during its generation. These accidents are a result of the susceptibility of the generating plants to component failure, human error, extreme environmental events, targeted attacks, and natural disasters. Given the complexity and high interconnectivity of the systems in question, a small glitch, otherwise known as an initiating event, could cascade to catastrophic consequences. It is, therefore, vital that the vulnerability of a plant to these glitches and their ensuing consequences be ascertained, to ensure that the appropriate mitigating actions are taken.

The reliability of a system is the likelihood that it survives a defined period and its availability is the likelihood of it being capable of performing its required functions on demand. These quantities are important to a nuclear power plant's safety because, a nuclear power plant by default is equipped with safety systems to inhibit the propagation of an initiating event. An accident ensues if the safety systems required to mitigate some initiating event are unavailable or incapacitated by the initiating event. It is, therefore, easy to see that the reliability, as well as the availability of these systems, shape the safety of the plant. These crucial quantities, currently, are estimated using legacy techniques like static fault and event tree analyses or their derivatives. Despite their popularity and widely acclaimed success, these legacy techniques lack the flexibility to implement fully the operational dynamics of the majority of systems. Most importantly, their ease of application deteriorates with increasing system size and complexity, such that the analyst is often forced to make unrealistic assumptions. These unrealistic assumptions sometimes compromise the accuracy of the results obtained and subsequently, the quality of the risk management decisions reached. Their inadequacy is often amplified if the system is composed of multi-state components or characterised by epistemic uncertainties, induced by vague or imprecise data. The ideal approach, therefore, should be sufficiently robust to not necessitate unrealistic assumptions but flexible enough to accommodate realistic system attributes, while guaranteeing accuracy.

This dissertation provides a detailed account of a series of computationally efficient system reliability analysis techniques proposed to address the limitations of the existing

probabilistic risk assessment approaches. The proposed techniques are based mainly, on an advanced hybrid event-driven Monte Carlo simulation technique that invokes load-flow principles to resolve, intuitively, the difficulties associated with the topological complexity of systems and the multi-state attributes of their components. In addition to their intuitiveness and relative completeness, a key advantage of the proposed techniques is their general applicability. They have been applied, for instance, to a variety of problems, ranging from the production availability of an offshore oil installation and the maintenance strategy optimization of the IEEE-24 bus test system to the probabilistic risk assessment of station blackout accidents at the Maanshan nuclear power plant in Taiwan. The proposed techniques, therefore, should influence robust decisions in the risk management of not only nuclear power plants but other critical systems as well. They have been incorporated into the open-source uncertainty quantification tool, OpenCossan, to render them readily available to industry and other researchers.

Declaration

I declare that this thesis was composed by myself and that the results contained herein have not been submitted for any other degree or professional qualification. I confirm that the work submitted is mine, except where it forms part of a jointly authored publication. In which case, my contribution and the names of the other authors have been explicitly indicated below. I confirm also that appropriate credit is given where reference is made to others' work and that the thesis contains 216 pages, 91 figures, and 30 tables.

The work in Chapter 7 is under consideration for publication in the *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* as "Extending the survival signature paradigm to complex systems with non-repairable dependent failures" with co-authors, G. Feng, F.P.A Coolen, M. Beer, and E. Patelli (supervisor). The framework was conceived by all the authors and I, in addition, carried out its algorithmic implementation and prepared the first draft of the manuscript.

10^{th} May 2018

List of Publications

Peer-reviewed Journal Publications

1. H. George-Williams and E. Patelli. A hybrid load-flow and event driven simulation approach to multi-state system reliability evaluation. *Reliability Engineering & System Safety*, 152:351-367, 2016. Available at `http://dx.doi.org/10.1016/j.ress.2016.04.002`

2. H. George-Williams and E. Patelli. Efficient Availability assessment of reconfigurable multi-state systems with interdependencies. *Reliability Engineering & System Safety*, 431-444, 2017. Available at `http://dx.doi.org/10.1016/j.ress.2017.05.010`

3. H. George-Williams and E. Patelli. Maintenance strategy optimization for complex power systems susceptible to maintenance delays and operational dynamics. *IEEE Transactions on Reliability*, 66(4):1309-1330, 2017. Available at `http://dx.doi.org/10.1109/TR.2017.2738447`

4. H. George-Williams, M. Lee, and E. Patelli. Probabilistic risk assessment of station blackouts in nuclear power Plants. *IEEE Transactions on Reliability*, 67(2):494-512, 2018. Available at `http://dx.doi.org/10.1109/TR.2018.2824620`

5. H. George-Williams, G. Feng, F.P.A Coolen, M. Beer, and E. Patelli. Extending the survival signature paradigm to complex systems with non-repairable dependent failures. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability* (Accepted).

Peer-reviewed Conference Publications

1. H. George-Williams and E. Patelli. Monte Carlo-based reliability/availability analysis algorithm for efficient maintenance planning. *In Proceedings of the Structural Mechanics in Reactor Technology Conference*, Vol. 23, Manchester, 2015.

2. H. George-Williams and E. Patelli. Efficient availability assessment of reconfigurable complex multi-state systems with interdependencies. *In Proceedings of the European Safety and Reliability Conference*, Vol. 26, Glasgow, 2016.

3. H. George-Williams, E. Patelli, and M. Lee. Reliability & performance analysis of multi-state systems based on analytical load-flow considerations. *In Proceedings of the European Safety and Reliability Conference*, Vol. 26, Glasgow, 2016.

4. H. George-Williams, E. Patelli, and M. Lee. A framework for power recovery probability quantification in nuclear power plant station blackout sequences. *In Proceedings of the Probabilistic Safety Assessment and Management Conference*, Vol. 13, Seoul, 2016.

5. E. Patelli, H. George-Williams, J. Sadeghi, R. Rocchetta, M. Broggi, M. de Angelis. OpenCossan 2.0: an efficient computational toolbox for risk, reliability, and resilience analysis. *In Proceedings of the 7th International Symposium on Uncertainty Modelling and Analysis*, Florianapolis, 2018.

Contents

Nomenclature

ABBREVIATIONS

AC	Alternating Current.
BDD	Binary Decision Diagrams.
BPM	Basic Parameter Model.
CCF	Common-Cause Failures.
CCG	Common-Cause Group.
CFT	Condition-based Fault Trees.
CM	Corrective Maintenance.
DC	Direct Current.
DFG	Dynamic Flow Graphs.
DFT	Dynamic Fault Trees.
DRBD	Dynamic Reliability Block Diagrams.
EDG	Emergency Diesel Generator.
EENS	Expected Energy Not Supplied.
FT	Fault Trees.
LOOP	Loss Of Offsite Power.
MC	Markov Chain.
MGL	Multiple Greek Letter.
MTTF	Mean Time To Failure
NP	Non-Polynomial time.
PM	Preventive Maintenance.
PN	Petri Nets.
RBD	Reliability Block Diagrams.
RG	Reliability Graphs.
SBO	Station Blackout.
SDP	Sum of Disjoint Products.
UGF	Universal Generating Function.

NOTATIONS

$\boldsymbol{A} - \boldsymbol{B}$	Elements in $\boldsymbol{A}$ but not in $\boldsymbol{B}$.
$\boldsymbol{B}\,(\Downarrow n)$	Next n rows of matrix $\boldsymbol{B}$.
$\lceil a \rceil$	Smallest integer greater than a.

$(\boldsymbol{A}, m \rightarrow n)$	Elements m to n of vector $\boldsymbol{A}$.
$(\mathbf{B}, i)$	i^{th} element of vector $\boldsymbol{B}$.
$(EENS)_{eff}$	Total system EENS.
$[a, b]$	Maintenance strategy based on regimes a and b.
$Exp\,(a)$	Exponential distribution with rate $1/a$.
$f(t)$Ⓢk	Generate k random samples from $f(t)$.
$G\,(a, b)$	Gamma distribution with shape and scale parameters a and b.
$Gu\,(a, b)$	Gumbel distribution with mean, a and standard deviation, b.
$L\,(C, R\,(\boldsymbol{n}^*, k))$	System loss corresponding to maintenance strategy k.
$LogN\,(a, b)$	Log-normal distribution with mean, a and standard deviation, b.
$min\,(\boldsymbol{B}, b)$	The least element of $\boldsymbol{B}$ greater than b.
$min\,(\boldsymbol{B})$	The least element of vector $\boldsymbol{B}$.
$numel\,(\boldsymbol{B})$	Number of elements in set/vector $\boldsymbol{B}$.
$R\,(\boldsymbol{n}^*, k)$	System reliability and performance indices for maintenance strategy k.
$size\,(\boldsymbol{B}, 1)$	Number of rows of matrix $\boldsymbol{B}$.
$U\,(a, b)$	Uniform distribution with bounds on a and b.
$u \sim (0, 1)$	Uniform random number between 0 and 1.
$Wb\,(a, b)$	Weibull distribution with scale and shape parameters a and b.
APM	Awaiting preventive maintenance state.
CPHM	Cost per Hour of Maintenance.
CS	Cold Standby state.
EC	Electricity Cost.
F	Failed state.
FMC	Fixed Cost per Maintenance team.
I	Idle state.
PF	Partial failure state.
S	Shut-down state.
SU	Start-Up state.
TM	Test/Maintenance state.
W	Working state.

SYMBOLS

α_{ij}	Transmission efficiency of link between nodes i and j.
$\beta_1^{\{k\}}$	Common failure mode of CCG k.
$\beta_2^{\{k\}}$	State rendering CCG k susceptible to CCF.
$\boldsymbol{\beta}$	Matrix of possible system component combinations.
$\boldsymbol{\delta}$	Set of components in shut-down.
$\boldsymbol{\eta}$	Vector of system node flows.
$\boldsymbol{\Gamma}$	System incidence matrix.
$\mho_i^+$	Set of nodes sending flow to node i.
$\mho_i^-$	Set of nodes receiving flow from node i.

$\boldsymbol{\mu}$	Vector of current capacities of system nodes.
$\boldsymbol{\nu}$	Shared/dedicated maintenance indicator vector.
$\boldsymbol{\Phi}$	System equality constraint matrix.
$\boldsymbol{\Pi}$	Matrix defining the size of each maintenance group.
$\boldsymbol{\Psi}$	Vector of system performance history.
$\boldsymbol{\rho}^{\{k\}}$	Set of type k components.
$\boldsymbol{\tau}$	Vector of next transition times of nodes.
$\boldsymbol{\tau}_{pm}$	Vector of next preventive maintenance due times of components.
$\boldsymbol{\tau}_{spare}$	Vector of component spares availability times.
$\boldsymbol{\Theta}$	System inequality constraint matrix.
$\boldsymbol{\theta}^{\{k\}}$	CCF probabilities for CCG k.
$\boldsymbol{\theta}_j$	Total set of components in maintenance group j $\left(\boldsymbol{\theta}_j^{\{cm\}} \cup \boldsymbol{\theta}_j^{\{pm\}}\right)$.
$\boldsymbol{\theta}_j^{\{cm\}}$	Set of components assigned to group j for CM.
$\boldsymbol{\theta}_j^{\{pm\}}$	Set of components assigned to group j for PM.
$\boldsymbol{\varsigma}$	Component sink index vector.
$\boldsymbol{\vartheta}_1$	Set of components repaired only while in shut-down.
$\boldsymbol{\vartheta}_2$	Set of components which PM is initiated only while in shut-down.
$\boldsymbol{A}$	System adjacency matrix.
$\boldsymbol{C}$	Component capacity vector.
$\boldsymbol{D}_i'$	Joint dependency matrix of node i.
$\boldsymbol{D}_i$	Dependency matrix of node i.
$\boldsymbol{e}$	System edge matrix
$\boldsymbol{h}_t$	Global set of components in maintenance queue.
$\boldsymbol{h}_{1f}$	Final content of $\boldsymbol{h}_1$ after normalization.
$\boldsymbol{h}_1$	Set of components in CM queue.
$\boldsymbol{h}_{2f}$	Final content of $\boldsymbol{h}_2$ after normalization.
$\boldsymbol{h}_2$	Set of components in PM queue.
$\boldsymbol{I}$	Indicator register for subsystems affected by the last node transition.
$\boldsymbol{lb}$	Vector of minimum flow across links.
$\boldsymbol{L}$	System capacity matrix.
$\boldsymbol{n}^*$	Combination of maintenance teams.
$\boldsymbol{R}$	Set of ranks of subsystems associated with a node.
$\boldsymbol{s}$	Set of source nodes.
$\boldsymbol{S}_\tau$	System survival signature.
$\boldsymbol{S}_i$	Set of nodes belonging to subsystem i.
$\boldsymbol{T}$	Transition matrix of component.
$\boldsymbol{t}$	Set of sink nodes.
$\boldsymbol{t}_{times}$	Set of sampled transition times.
$\boldsymbol{ub}$	Vector of maximum flow across links.

$\boldsymbol{V}$	Set of system nodes.
χ	Number of power trains generator can supply simultaneously.
δ	System time step.
δ_t	Total number of system time steps.
η_i	Flow through node i.
$\eth$	Number of intermediate nodes.
Λ	Minimum load rating of component.
λ_1	Total number of busy corrective maintenance teams.
λ_2	Total number of busy preventive maintenance teams.
λ_t	Total number of busy maintenance teams.
$\lambda_j^{\{cm\}}$	Number of busy CM teams in group j.
$\lambda_j^{\{pm\}}$	Number of busy PM teams in group j.
λ_{m-n}	Failure rate from state m to n.
$\mathbb{C}$	Cascading matrix.
$\mathbb{D}_i$	Set of dual nodes of node i.
$\mathbb{E}$	Set of component properties.
$\mathbb{E}_i$	Properties of component i.
$\mathbb{F}$	Set of failed components/nodes.
$\mathbb{H}$	Matrix of CCF probabilities.
$\mathbb{H}'$	Cumulative sum of $\mathbb{H}$ along rows.
$\mathbb{I}_i$	Set of components which failure is induced by component i.
$\mathbb{L}_i$	Load dependency parameter of node i.
$\mathbb{N}$	Set of all possible maintenance team combinations, $\{\boldsymbol{n}_1^*, \boldsymbol{n}_2^*, ..., \boldsymbol{n}_\phi^*\}$.
$\mathbb{O}$	Objective function.
$\mathbb{T}$	Vector of system transition times.
$\mu_i^{\{cm\}}$	Indicator function for CM suspension of component i.
$\mu_i^{\{pm\}}$	Indicator function for PM suspension of component i.
μ_{m-n}	Repair rate from state m to n.
ω	Total number of system maintenance groups.
Ω_{ij}	Maximum flow from node i to node j.
$\underline{\boldsymbol{X}}$	Set of system state vectors corresponding to $\underline{\boldsymbol{x}}'$.
$\underline{\boldsymbol{x}}$	System state vector.
$\underline{\boldsymbol{x}}'$	Modified system state vector.
$\underline{\boldsymbol{x}}_k$	State vector for group of type k components.
ε_x	Sink index of component in state x.
$\varepsilon_x^{\{i\}}$	Sink index of component i in state x.
$\varphi(\underline{\boldsymbol{x}})$	System structure function.
ξ	Set containing the size of each component group.
$A(t)$	Instantaneous system availability.
c_x	Current capacity of component.

$C_i^{\{cm\}}$	Cost per hour of corrective maintenance of component i.
$C_i^{\{pm\}}$	Cost per hour of preventive maintenance of component i.
c_{max}	Maximum capacity of component.
$C_{s_i}^{\{cm\}}$	Unit cost of CM spares for component i.
$C_{s_i}^{\{pm\}}$	Unit cost of PM spares for component i.
$c_{x^-}^{\{i\}}$	Capacity of node i before state transition.
$c_x^{\{i\}}$	Current capacity of component i.
e_{ij}	Link/edge between nodes i and j.
f_l	LOOP frequency.
f_s	SBO frequency.
$f_{xy}(t)$	Probability density function of transition time from state x to y.
G	System graph model.
K	Number of component types.
k	Number of edges in system graph.
k_f	Multiplication factor.
k_i	Proportion of PM duration of component i spent before interruption.
l_{ij}	Capacity of link/edge between nodes i and j.
M	Total number of system nodes.
m	Total number of power trains.
M'	Total number of maintainable components.
M''	Number of external nodes.
m_j	Number of components assigned to maintenance group j.
M_k	Number of type k components.
N	Number of simulation samples.
n	Number of component states.
n_1	Total number of corrective maintenance teams.
n_2	Total number of preventive maintenance teams.
n_t	Total number of maintenance teams.
n_{1_j}	Number of corrective maintenance teams in group j.
n_{2_j}	Number of preventive maintenance teams in group j.
$N_i^{\{cm\}}$	Number of successful corrective maintenance actions on component i.
$N_i^{\{pm\}}$	Number of successful preventive maintenance actions on component i.
n_{t_j}	Total number of maintenance teams in group j.
$P(\underline{\boldsymbol{x}}')$	Occurrence probability of $\underline{\boldsymbol{x}}'$.
$p_1^{\{sbo\}}$	Conditional SBO probability given LOOP.
$p_n^{\{sbo\}}$	Conditional n^{th} SBO probability given the $(n-1)^{th}$ SBO.
$p_i^{\{s\}}$	Probability of spares being used in the repair of component i.
p_z	Average probability of state z.
$p_z(t)$	Instantaneous probability of state z.
$q_i^{\{s\}}$	Probability of spares being used in the PM of component i.

$r'(t)$	System non-recovery probability.
$r'_2(t)$	Non-recovery probability from second SBO.
$r(t)$	System recovery probability.
$R(t)$	System reliability.
s_j	SBO indicator for j^{th} simulation sample.
$s_i^{\{cm\}}$	Number of spares used for corrective maintenance of component i.
$s_i^{\{pm\}}$	Number of spares used for preventive maintenance of component i.
t'	Maximum remaining lifetime of a component in APM state.
T_m	Mission time.
$t_i^{\{cm\}}$	Time spent by component i in corrective maintenance.
$t_i^{\{pm\}}$	Time spent by component i in preventive maintenance.
t_{next}	Next transition time.
t_{pm}	Component expected preventive maintenance duration.
t_{sample}	Minimum sampled transition time.
t_{spent}	Time spent in shut-down.
u	Proportion of power train demand demand generator satisfies.
U_{tm}	Unavailability due to test or maintenance.
x	Current state of component.
x'_k	Number of available type k components.
$X(t)$	Expected instantaneous system output.
x_0	Initial state of component.
x_i	State of i^{th} component.
x_s	State of system.
X_{ij}	Flow across link/edge e_{ij}.
y'	Next failure state of a component in APM state.
y_m	Next maintenance state.
y_{next}	Next state of component.

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

A system is a collection of entities and/or processes working in unison to achieve a common goal. The likelihood of this goal being attained, given a certain operating condition, defines the reliability of that system. One can deduce from this definition that system reliability is relative rather than absolute. It depends on the operating condition of the system, the criteria against which mission success is determined, and the mission duration. For example, a car with a maximum achievable speed of 100km/h would be deemed perfectly reliable for a mission requiring 1000km be covered in 12 hours, barring component failures. The same car, however, under the same conditions, would be absolutely unreliable if the mission were to be completed under 10 hours. Again, if the operating condition were modified to regard component failures, it would be impossible to be certain about mission success, without a detailed and representative reliability analysis of the car. This example highlights the relativity of system reliability. Strictly, a system, as defined earlier, may refer to a biological, financial, or an engineering system. For the purpose of this thesis, however, it refers to the last of these, and is a collection of mechanical and electronic components. Structural systems like bridges and buildings, which may also be regarded as engineering systems, are outside the scope of this thesis.

An engineering system can be classified along several parameters. For instance, on the basis of its operational period, a system is either continuous or mission-oriented. A mission-oriented system, like a rocket taking essential supplies to the international space station, performs a mission of fixed duration [145]. Continuous systems, on the other hand, have an infinite mission duration, and their continuity is limited only by their lifespan. Prominent examples include power plants, transportation networks, water distribution systems, and power grids. These systems can also be repairable or non-repairable. Unlike repairable systems, non-repairable systems are built of components that cannot be repaired within the mission. Repairable components are mostly found in

continuous systems and, in fact, are the very reason the systems sustain their continuity. There are many more classifications of systems, but these are extensively dealt with in Chapter 2 and the subsequent sections of this chapter.

1.2 Complex System Reliability

A system can be classed as complex from two fronts: complexity in terms of the functional relationships between its components, and complexity due to its structure/topology. A system is deemed structurally complex if it is a non-series, non-parallel, or non-series-parallel interconnection of components. The components, which are the system's smallest building block, determine its output levels, states, and behaviour. In realistic systems, these components may exist in one of several possible states/output levels [155], dictated by their failure characteristics, operating conditions, age, or some stochastic event outside the system boundary. The result is a system characterised by multiple states, with the number of states determined by the diversity in the states of the components, as well as the structure of the system [38, 75].

Unlike binary-state systems which can only be perfectly working or completely failed, multi-state systems can exist in intermediate states, as well. The number of intermediate states may or may not be finite, depending on the performance measure under consideration and the type of system [75]. For instance, the power generated by a hydroelectric power plant may take any value between zero and its maximum achievable value, depending on the height of water in the dam, the performance levels of its components, and the demand on the grid. Other examples of multi-state systems are communication systems; where data processing speed [78, 87] may be the performance measure, cooling systems; where coolant flow rate or cooling capacity [51] may be the performance measure, and production systems; where production rate is the performance measure. These systems may be standalone or form an indispensable part of some critical system like safety-critical and industrial control systems. It is, therefore, important to be able to assess their susceptibility to failures, quantify, and predict the ensuing consequences, for effective planning of preventive and corrective measures.

Reliability prediction transcends just defining a set of standards for predicting the failure rate of components to system-level and the safety of complex systems [123]. It has phased through tremendous developments, moving from traditional methods treating the failure of a system as a consequence of the failure of its components only, to methods that approach its failure from a wider perspective, including external factors [34].

1.3 Motivation

Failure is an inherent phenomenon in mechanical and electronic components and systems. It is a consequence of their material properties, operating condition, and age.

This highlights the impossibility to infinitely operate a system, without scheduled and unscheduled outages. Scheduled outages are due mainly to preventive maintenance and inspections but should be planned in a way that inflicts minimal disruptions to the operation of the system. Even though the system operator has no control over when unscheduled outages occur, they can still institute measures to reduce their frequency, mitigate their effects, and ensure the timely recovery of the system when they occur.

An engineering system may be as simple as the smallest circuit in a transistor radio or as critical and complex as an entire nation's transportation, power, communication, and water network. One thing is certain, however, regardless of the size and criticality of a system, its failure is often accompanied by consequences. These consequences range from a mere discomfort (like the one one feels when one's air conditioner is broken) to the more severe phenomena of economic loss and the loss of human lives. In May 2017, for instance, a power failure crippled British Airway's (BA) IT system at the London Heathrow and Gatwick airports in the United Kingdom [21]. The outage, which lasted only a few days, caused hundreds of flight cancellations, leaving about 75,000 passengers stranded, and costing the airline £58m [22]. Similarly, in April 2010, the explosion of the Deepwater Horizon drilling rig leased to British Petroleum (BP) in the Gulf of Mexico killed 11 workers, triggering an oil spill believed to be the worst in US history [17]. The disaster did not only bring financial losses to the company, it created a big dent on its reputation too. In fact, BP were forced to put aside $41bn (more than twice their profit in 2009) to cover clean up costs and legal fees [18]. The reliability analysis, therefore, of engineering systems, is very important, since it is the only way their susceptibility to failures and the ensuing consequences can be quantified.

There are a couple of other reasons why one would want to compute the reliability of a system. At the design stage, for instance, the design engineer would want to ensure a design meets the desired performance and safety requirements or to select the best of multiple designs. After commissioning, a detailed reliability analysis of a system can confirm whether or not it performs as expected and can reveal any vulnerabilities, which information is useful for more robust future designs. There is also maintenance strategy optimization, where a complete reliability analysis of a system is required to compute the benefits of possible strategies, and hence, deduce the best maintenance strategy.

1.3.1 The Role of System Reliability Analysis in Nuclear Safety

Nuclear power plants are a typical example of a system that demands the highest reliability of even its smallest component. Some failure events do not only result in their loss of output but sometimes set off a sequence of events with far-reaching consequences as well. The science of examining what can go wrong in a plant and its ensuing risks is known as probabilistic risk assessment (PRA). It entails the identification of the events (known as initiating events) that could trigger unwanted consequences and the

subsequent computation of their frequencies and extent [137]. For nuclear power plants, the primary unwanted consequences are core damage, reactor containment breach, and the release of radioactive materials into the environment. Though these events are extremely unlikely, there have been several nuclear accidents in history, some of which dates to the 1950s [19]. Of these accidents, only the three infamous ones (the Three Mile Island, Chernobyl, and Fukushima Daiichi disasters), however, are explored in this thesis, with focus on their causes, consequences, and similarities.

1.3.1.1 The Three Mile Island Accident

Three Mile Island was a 2-unit pressurised water reactor nuclear power plant in Dauphin county, Pennsylvania. On the 28th day of March 1979, a failure of one of its cooling systems initiated a sequence of events that would later be regarded the most serious nuclear incident in commercial nuclear power generation history in the United States [136]. Despite its relatively minor health and environmental consequences, however, its aftermath sparked a revolution in the operation, regulation, and emergency response protocols of nuclear power plants in the United States. The accident, rated 5 on the 7-point international nuclear events scale, was initiated by the failure of the main feedwater pumps in the second reactor, which was later traced to imperfect maintenance [62]. Feedwater pumps supply cooling water to the steam generator and maintain circulation in the secondary cooling loop of a nuclear reactor. When these pumps failed, the unit's turbine-driven generator and reactor automatically shut down, and the core starved of cooling, began to record a rapid temperature rise. This, in turn, resulted in pressure building up in the reactor, which further actuated the pressure relief valve. However, the valve, which was to reset itself when the pressure fell to an acceptable level, remained stuck open, leading to a loss of coolant accident. Even when the reactor incident alarms went off, the operators did not know a loss of coolant accident was already in progress. In consequence, they reduced coolant circulation in the primary cooling loop, which fueled the subsequent partial meltdown of the reactor, due to excessive temperatures [136]. The reactor containment, however, remained intact, and there were no substantial releases of radioactive materials into the surrounding, save for a small amount of radioactive gases in the vicinity of the plant a few days later. These small releases were due to the loss, through the stuck-open valve, of coolant contaminated with radionuclides.

It is clear the accident was initiated by component failure (the failure of the feedwater pumps and the stuck-open valve) but its progression was exacerbated by design deficiencies, human error, and organisational & quality problems [62, 136]. For instance, even when the valve was stuck open, the control instrumentation indicated a closed valve. The operators, prior to the incident, were aware that the valve leaked but postponed its maintenance. Both the valve failure and false indication, therefore, would have been prevented by effective design and maintenance policy reconsiderations.

1.3.1.2 The Chernobyl Nuclear Disaster

The Chernobyl nuclear power plant was a 4-unit graphite-moderated light water reactor, in the Ukrainian city of Chernobyl. Occurring on the 26th of April 1986, the accident, which is arguably the world's worst nuclear disaster, resulted from an experiment-gone-wrong culminating into the explosion of the fourth reactor. The disaster is believed to have released about 2 orders of magnitude more radiation than the atomic bombs dropped on Hiroshima and Nagasaki. More than 350,000 people were resettled and traces of radioactive materials from the accident were found almost everywhere in the northern hemisphere [16]. Two deaths were recorded on the plant premisses, as a direct consequence of the blasts and a further 28 deaths, from exposure to radiation [102].

The accident is attributable to human errors during an experiment to investigate the ability of the fourth reactor to drive its cooling pumps at low power. Executing their plans, the engineers had inserted the control rods into the reactor core, with the view to reducing its power only to about 20% of its nominal value. Too many control rods, however, were introduced, such that the reactor was almost shut down by Xenon poising [129]. In response, the engineers withdrew some rods from the reactor, and in two hours, managed to stabilize the reactor power at about 12% and commenced the test. However, too many rods were withdrawn, and less than 1 minute into the test, the reactor power shot up to about 100 times its nominal value [16]. The reactor's automatic shut-down system which had been disabled to allow the low power operation of the reactor, was reactivated. Its reactivation led to the insertion of more control rods into the core, displacing the mixture of steam and hot coolant that had built up. The result was two catastrophic explosions, blowing away the roof of the reactor building.

Unlike the Three Mile Island accident, component failure had no part in the Chernobyl disaster. They, however, share human error and design deficiencies, as root causes. In fact, the experiment was scheduled to have been performed by the day shift, who had been trained for the exercise. The night shift, who had very little time to prepare, had to take over after the initial schedule was disrupted by some eventuality. Secondly, the young engineer who was assigned to the control rods had been working independently for only three months, and may have initiated the accident when he erroneously inserted the control rods too deep. Finally, the consequences of the accident may have been minimized if the reactor were housed in a concrete containment, as present-day reactors are. This was a design inadequacy of the Chernobyl nuclear power plant.

1.3.1.3 The Fukushima Daiichi Disaster

Unlike the Three Mile Island and Chernobyl disasters, the Fukushima accident was triggered by a natural disaster. The accident ensued from a magnitude 9.0 earthquake-triggered 40.5m high Tsunami in east Japan, on the 11th of March 2011 [129]. The tsunami reached about 14m at the 6-unit plant, whose units 4-6 were not in operation

at the time. As a result of the earthquake, the operational reactors automatically shut-down, at which time the backup generators were to take over the powering of the safety systems. These generators, however, were housed in the basement of the turbine building, which was only protected against a 10m flood height. Consequently, the generator room was flooded, initiating a station blackout sequence, culminating into a series of hydrogen explosions that damaged reactors 1-4 [129]. Since the reactors were housed in a concrete containment, the explosions only affected the reactor buildings. The unusual levels of radioactivity recorded in the vicinity of the plant were due to the venting operation employed to relieve the pressure in the respective reactor vessels.

No deaths were linked to the direct exposure to the radioactive nuclides released, but the accident eventually cost the plant operator billions of dollars in compensation claims. Though triggered by natural events, the progression of the accident was largely down to organisational and regulatory deficiencies. In fact, the chairman of a Japanese parliamentary panel set up to investigate the accident, Kiyoshi Kurokawa, called it a manmade disaster [20]. The operator had ignored earlier safety concerns about the layout of the emergency cooling system and the vulnerability of the plant to tsunamis. There was no robust probabilistic risk assessment of the plant and the operator's emergency response plan was lacking, the panel found. It is even alleged that relevant sections of the severe accident instruction manual were missing [20].

1.3.1.4 Concluding Remarks

From the preceding reviews, one could argue that nuclear accidents result from at least one of three causes: component failure, natural disasters, and human factors. The human factors may include, but are not limited to, human errors, targeted attacks, negligence, and procedural breaches. Given the unpredictability of these root causes and the potential severity of an accident, probabilistic risk assessment is key to the safe operation of nuclear power plants.

Probabilistic risk assessment in nuclear power plants is decomposed into three levels [137]. Level 1 PRA identifies all those events with the potential to inflict core damage and estimates the total core damage frequency of the plant. It models the response of the plant's safety systems to accidents and involves, mostly, the reliability analysis of these systems. Level 2 PRA, on the other hand, assumes core damage and investigates the quantity of radioactive materials released and how soon, after core damage, this begins. The reliability analysis of the concrete containment of the reactor forms the basis of the analysis here. Level 3 PRA estimates the consequences of the release on life, the environment, and the economy, constitutes level 3 PRA. System reliability analysis, therefore, as could be deduced from the preceding, is the backbone of probabilistic risk assessment in nuclear power plants.

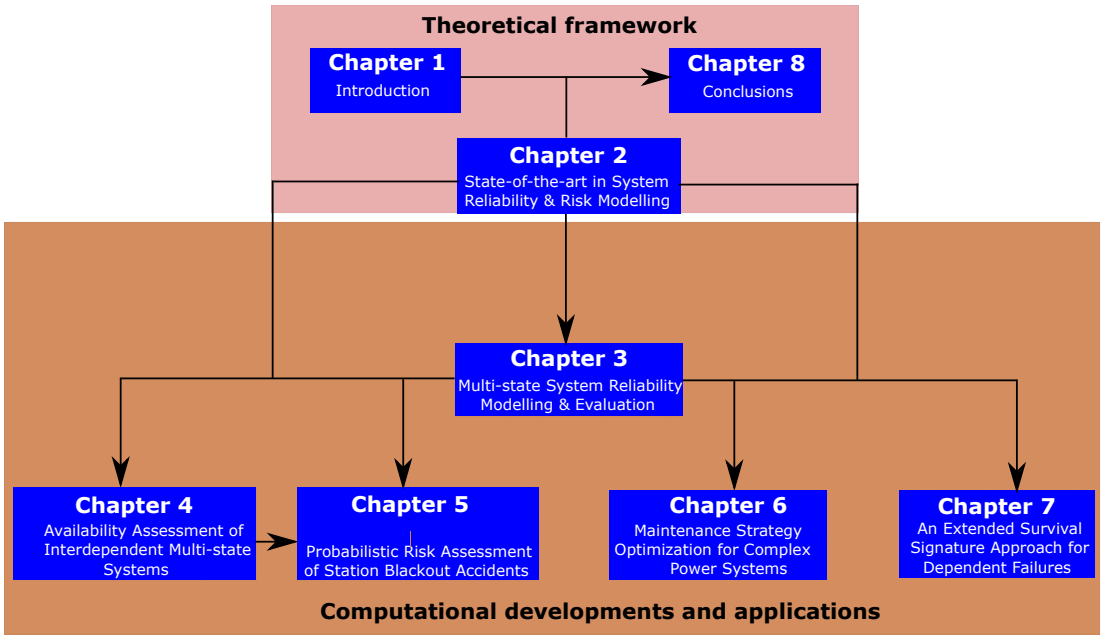


Figure 1.1: Thesis structure and relationships between chapters.

1.4 Aims and Objectives

Computing the reliability of a realistic system is not always simple. Even with the existence of a number of computational techniques, there still is a significant room for improvement. The aim, therefore, of this thesis is to develop an approach or a series of approaches to simplify, as well as enhance the reliability analysis of realistic engineering systems, without the need for unrealistic assumptions. The thesis, specifically, will seek:

1. to develop an intuitive means of resolving difficulties arising from the topological complexity of systems during their reliability analysis;
2. to develop component and process modelling techniques capable of replicating reality as close as possible, with the view to obtaining credible reliability estimates;
3. to intuitively model all forms of inter-component dependencies in systems;
4. to improve maintenance modelling and optimization in practical systems;
5. to apply the computational tools and methodologies developed in 1-4, to the probabilistic risk assessment of nuclear power plants.

1.5 Thesis Structure

This thesis comprises eight (8) chapters, organised in two (2) groups, as shown in Figure 1.1. The first group, composed of Chapters 1, 2, and 8, provides the necessary theoretical framework for an understanding of the thesis and the research questions

it seeks to answer. Chapter 2, for instance, reviews the existing system reliability modelling techniques and their applicability to complex system reliability evaluation, maintenance modelling & optimization, and the probabilistic risk assessment of nuclear power plants. The extent to which the thesis answers the research questions asked in Section 1.4 and addresses the inadequacies of existing techniques identified in Chapter 2, is discussed in Chapter 8, which also offers recommendations for future research.

Each of the remaining chapters builds on the inadequacies identified in Chapter 2 to develop a computational approach and showcase its applicability via a case study. In this light, Chapter 3 proposes an intuitive load-flow Monte Carlo simulation technique to resolve the difficulties emanating from the topological complexity, as well as the multi-state behaviour of systems. This approach is extended to interdependent systems in Chapter 4, and used to assess the production availability of an offshore oil installation characterised by limited maintenance teams. Chapter 5 generalises the cascading failure model proposed in Chapter 4 and in conjunction with the load-flow simulation proposed in Chapter 3, proposes an approach to simulate station blackout accidents in nuclear power plants. The chapter concludes with the probabilistic assessment of station blackout risks at the Maanshan nuclear power plant in Taiwan.

The case study presented in Chapter 4 highlights the need for a general procedure for the maintenance modelling of multi-state systems in the presence of maintenance delays and operational uncertainties. Consequently, such a maintenance modelling approach is proposed in Chapter 6, complemented by an efficient algorithm, proposed to identify the best maintenance strategy. Being mindful of the demanding computational intensity of simulation-based techniques, Chapter 7 extends the efficient survival signature approach to systems susceptible to cascading and common-cause failures, comparing the resulting framework with the load-flow simulation approach proposed in Chapter 3.

Chapter 2

The State of the art in System Reliability & Risk Modelling

2.1 Existing Reliability Modelling Techniques

In system reliability and performance evaluation, the analyst has numerous techniques at their disposal. Sometimes, one technique cannot quite yield the required outcome, and a collection of techniques is required, instead. The technique employed is determined by the system being analysed, the reliability indices of interest, the available computing resources, and the degree of precision demanded. These techniques, according to [1,128] (cited in [38]), can be classed as heuristic, analytical, or simulation-based. They can also be classified on the basis of applicability, in which case they can be static or dynamic [38]. Unlike static techniques, dynamic techniques do not only model the system based on the functional and structural relationships between its components, but also support dynamic relationships like inter-component dependencies.

Reliability Block Diagrams (RBD) [11,144] and Fault Trees (FT) [11,141] have been used extensively, in the reliability evaluation of binary-state systems. RBD are a graphical expression of the functional relationships between system components in terms of the combination of functioning components required for system success. FT, on the other hand, express this functional relationship via boolean logic gates and depict the combination of component failures that culminate in system failure. Both techniques have proven particularly useful for moderately sized systems with series-parallel configurations. However, they become difficult to apply with large or complex systems and often require additional techniques [4] to decompose the system. Overcoming this difficulty necessitated the development of Reliability Graphs (RG) [113, 147]. RG represent the system as a network of nodes connected by edges and failure is defined as the non-existence of path between the source and sink nodes [38]. They are very much efficient in modelling structural complexities.

RBD, FT, and RG assume components to be statistically independent, rendering them inadequate for systems susceptible to restrictive maintenance policies and component interdependencies. However, techniques including but not limited to Dynamic Reliability Block Diagrams (DRBD) [38], Dynamic Fault Trees (DFT) [24], Condition-based Fault Trees (CFT) [126], Dynamic Flow Graphs (DFG) [3], Petri Nets (PN) [95], and other combinatorial techniques [138] have been developed to model these dynamic relationships. They have found application in a wide range of reliability engineering problems, including repairable systems with restrictive maintenance policies [5,6,95,138].

Though the earliest forms of these techniques were applicable only to binary-state systems, numerous instances of their recent extension to multi-state systems exist. Lisnianski [88], for instance, employed an extended block diagram method to apply classical block diagram principles to a repairable multi-state system. Binary Decision Diagrams (BDD) [146,156], whose underlying principles are built upon Boolean algebra, have also been applied, with success, to multi-state system reliability evaluation. They proceed via a state enumeration procedure in which each system state is represented by a multi-state fault tree. Unfortunately, state enumeration is only feasible for moderately sized simple systems. For large/complex systems, it is expensive and error-prone when done manually, which limits the applicability of BDD to moderately sized simple systems.

In recent years, RG have also attracted significant attention, which interest has given birth to algorithms optimized for multi-state system reliability evaluation. Despite Yeh's successful attempts, on two separate occasions ([148] and [150]), of developing algorithms that do not require prior knowledge of all minimal paths or cuts of the system, most graph-based algorithms [83, 84, 151, 162] do. Therefore, exploiting them requires first deriving the desired path or cut sets, using other well-known algorithms, which is an NP-hard problem [148, 150]. Compounding the challenges of these graph-base algorithms, including Yeh's algorithms in [148] and [150], is the fact that they are based on the assumption that the capacities of system components are integer-valued. However, in many systems, components and system capacities are not necessarily integer-valued.

In addition to their individual shortfalls, the extended block diagram technique, BDD, and graph-based algorithms share two common limitations. First, they define reliability with respect to the maximum flow through the system. Therefore, they are limited to systems with single output nodes or systems (like signal transmission networks [73]) with multiple output nodes in which only the presence of flow at these nodes is desired but the relative magnitude is irrelevant. They stop short at solving multi-output systems with competing demand at the output nodes. The second limitation arises from the assumption that there are no flow losses in the system, making them inapplicable to certain practical engineering systems in which flow under some condition (e.g. component failure) escapes across the component/system boundary.

Various researchers [51, 78, 79, 87, 153, 155] have made invaluable contributions to multi-state system reliability analysis, developing techniques applicable to a wide range

of systems. These techniques have mainly been based on either the structure function approach, stochastic process, simulation, or the Universal Generating Function (UGF) approach [75, 77, 89]. The most popular stochastic process employed in reliability analysis is the Markov Chain (MC), which involves enumerating all the possible states of the system and evaluating the associated state probabilities [89, 143]. This technique is only easily applicable to exponential transitions or distributions with simple cumulative distribution functions, requires complicated mathematics, and becomes complex for large systems. The number of states in the model ranges from $M + 1$, for binary-state series systems, to 2^M, for binary-state parallel systems, M being the number of system components. For large multi-state systems, the number of states increases dramatically, rendering the model difficult to construct and expensive to compute.

The UGF was introduced to address the state explosion problem of the MC. It allows the algebraic derivation of a system's performance from the performance distribution of its components [75, 86]. However, both the UGF and MC are limited in the number of reliability and performance indices they can quantify. In [85], Lisnianski introduced the L_z-Transform and the inverse L_z-Transform concepts to enhance the derivation of instantaneous reliability measures like, the system reliability, the instantaneous availability, and the instantaneous output, with the UGF [51, 86]. The UGF is a powerful tool, its applicability has been extended even to systems with dependent components, as illustrated by Levitin [76, 77]. However, like all multi-state system reliability evaluation techniques, the UGF is maximum-flow-based and assumes flow conservation across components. Also, though straight-forward for systems with simple series/parallel architecture, it requires substantial effort for complex topology systems. These considerations have hindered its application to certain multi-state systems.

Simulation techniques [125, 161] are the most suitable for multi-state system reliability and performance evaluation, since they mimic the actual operation of the system. Their advantages over other techniques are derived from the fact that they can support any transition distribution, allow the effects of external factors on system performance to be investigated [161], and are easily integrated with other techniques [57, 130]. Though computationally expensive for large systems and small failure probabilities, techniques that reduce the computation time and effort now exist, thanks to recent advances in computing. Variance reduction techniques and parallel computing can reduce the computation time and effort by substantial amounts when adopted. Also in extensive use are Subset Simulation [8] and Line Sampling [32, 93], both improving the efficiency of simulations. Inspite of their potential, however, some of the existing simulation approaches [61, 81, 161] require prior knowledge of the system's path set, cut set, or structure function. Even those, like the one proposed by Yeh et al. [152], which do not require knowledge of these parameters, are applicable to binary-state systems only.

2.2 Dependencies in Engineering Systems

Engineers and system designers are under immense pressure to build systems robust and adequate enough to meet the ever increasing human demand and expectation. Unavoidably, the resulting systems are complex and highly interconnected, which ironically constitute a threat to their resilience and sustainability. The majority of the systems we interact with exist as multi-state interdependent systems. Two systems are interdependent if at least a pair of components (one from each system) are coupled by some phenomena, such that a malfunction of one affects the other. The coupling phenomenon could be proximity in space [23], functional dependence/interdependence [159], or both [158]. A water distribution network, where pumps and other electrical power-driven appliances rely on the reliability and performance of the power grid is a typical example.

The components of a system are normally prone to random failures arising from their intrinsic properties or induced failures stemming from targeted attacks [13], extreme environmental events [2], or erroneous human-system interactions. In interdependent systems, an undesirable glitch in one system could cascade and cause disruptions in coupled systems. The cascade could be fed back into the initiating system and the overall consequences may be catastrophic [23,63]. This was made clear by the massive blackout that struck Italy in September 2003, affecting the internet network in the process. In the same year, North America was hit by a 4-day blackout, affecting parts of USA and Canada [135]. To minimize the effects of failures, some interdependent systems are equipped with reconfiguration provisions. This normally entails transferring operation to another component, rerouting flow through alternative paths, or shutting down parts of the system. It is, therefore, vital to analyse the system's performance under the spectrum of possible vulnerability conditions, for adequate planning of defences [160].

In general, the achievement of maximum overall system performance is desirable. However, in many applications, it is more important to recover the required system performance in the shortest possible time, after component failure. This is the case, for instance, in nuclear power plant risk assessment, where the time-dependent recovery probability of offsite power is an important input to the overall safety of the plant [53]. Hence, system recovery time is not only a performance parameter, but a fundamental safety parameter, as well. Given the positive correlation between costs and resources (human, financial, and material) required to maintain a system, under economic constraints, there may not be sufficient resources for a speedy recovery. Therefore, an informed and robust decision making process would dictate that the decision support tool used be capable of modelling the relevant realistic aspects of the system, including the possibility of limited recovery response.

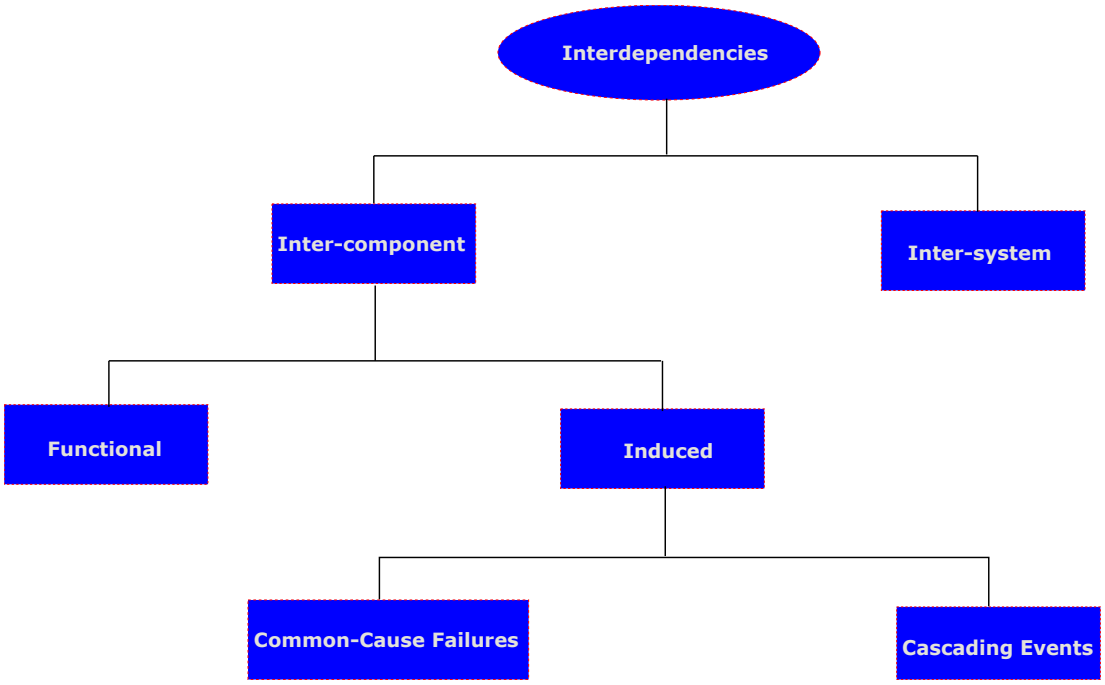


Figure 2.1: Forms of interdependencies in engineering systems.

2.2.1 Forms of Interdependencies

Interdependencies in engineering systems are manifested at two levels: between components (inter-component), which can be functional or induced and between systems/subsystems (inter-system). Functional dependencies are due to the topological and/or functional relationships between components. For instance, a motor-operated valve would not work if the electric motor controlling its actuator stopped due to a breaker failure. In this case, the valve is said to be functionally dependent on the breaker through the motor. Induced dependencies, on the other hand, are due to a state change in one component (the initiator) triggering a state change in another (the induced), such that even when the initiator is reinstated, the induced does not reinstate, unless manually made to do so. In the valve-motor-breaker example, for instance, the valve would resume its normal operation once the faulty breaker is replaced, highlighting the dichotomy between functional and induced dependencies. Functional dependencies are intrinsically accounted for by the innate attributes of the system reliability modelling and evaluation technique while induced dependencies require explicit modelling. Induced dependencies are further divided into Common-Cause Failures (CCF) and cascading events, as illustrated in Figure 2.1.

Inter-system dependencies are due to functional or induced couplings between multiple systems. Unlike standalone systems, functional dependencies in these systems may require explicit modelling. This is the case especially for components relying on material generated and transmitted by those of another system, under which condition the

reliability modelling technique used may prove inadequate.

2.2.1.1 Common-Cause Failures

Common-Cause Failures (CCF) are the simultaneous failure of multiple similar components due to the same root cause [103–105]. Their origin is traceable to a coupling that normally is external to the system. Notable instances are shared manufacturing lines/materials, shared maintenance teams, shared environments, and human error. A group of components susceptible to the same CCF event is called a Common-Cause Group (CCG). An important point to note about Common-Cause Failures is that, on occurrence of the failure event, there is a probability associated with multiple component failure and that the affected components fail in the same mode. Consequently, the number of components involved in the event ranges from 1 to the total number of components in the CCG. CCF events may affect an entire system or only a few of its components. They have been shown (in [36], for instance) to decrease the reliability and performance of multi-component systems. They, therefore, must be given due consideration in system reliability evaluation, to minimise overestimation.

CCF modelling and quantification has always attracted keen interest from both researchers and practitioners of system reliability and safety engineering. A total of five parametric models have been put forward to express the CCF probability associated with a CCG. The original model, the Basic Parameter Model (BPM), expresses the probability of a basic failure event involving a specific number of components. The other models, the β-factor model, the Multiple Greek Letter Model (MGL), the α-factor model, and the Binomial Failure Rate model, are mere reparameterizations of the BPM. Of these, the MGL and the α-factor models are the most widely used in system reliability and risk assessment. See Refs. [103, 104] for details on these models and their relationships.

Rasmuson and Kelly reviewed in their work [116], the basic concepts of modelling CCFs in reliability and risk studies. Rausand and Arnljot [117] proposed the square-root method, a simple bounding technique that estimates the effects of CCF on a system but which, however, lacks a strong mathematical foundation to support its application to practical systems. A robust Bayesian approach for quantifying the α-factor parameters of a CCG in the presence of epistemic uncertainties has also been put forward by Troffaes et al. [133]. Their approach, however, is limited to component-level reliability and, therefore, requires a second approach to obtain the system-level reliability indices. For this, Fan's stochastic hybrid systems model [48], O'Connor's general cause-based methodology [106], or Ramirez-Marquez's reliability optimization approach [114], amongst others, would do, and only if the reliability analyst is willing to turn a blind eye to their respective drawbacks - these models are built on reliability evaluation techniques that do not segregate the topological from the probabilistic attributes of the

system. As such, they are computationally expensive for problems involving multiple reliability analysis of the same system. They also have yet to be applied to multi-state systems, as well as systems susceptible to both cascading and common-cause failures.

2.2.1.2 Cascading Failures

Cascading failures are those with the capacity to trigger the instantaneous failure of one or more components of a system. They can originate from a component or from a phenomenon outside the system boundary. The likelihood of the initiating event originating from within the system, distinguishes them from CCF. Another point of dichotomy is that the affected components do not necessarily have to be similar or fail in the same mode. In addition, at the occurrence of the initiating event, the probability of all the coupled components failing is unity, save for the case when they are in a state rendering them immune. A few prominent examples of initiating events external to the system are extreme environmental events, natural disasters, external shocks, erroneous human-system interactions, and terrorist acts.

Various models have been developed to study the effects of cascading failures on complex systems [108, 160]. However, a good number of these models only assess their response to targeted attacks, variation in some coupling factor, or the relative importance of system components [23, 121, 159]. According to Ouyang [108], these models alone cannot sufficiently analyse the performance of interdependent systems. He intimated that flow based approaches, taking into account material or service flow across the system were required. When faced with the additional situation of random component failures, a complete reliability and availability analysis should be performed. However, renowned analytical multi-state system reliability evaluation techniques like BDD [146, 156], Sum-of-Disjoint-Products (SDP) [151], and UGF [75, 77, 89] are of very little use to the evaluation of these systems. Their inapplicability is amplified if, components can undergo non-Markovian transitions, their restoration can be delayed, the system is reconfigurable, or in the case of BDD and SDP, the system is complex, such that state enumeration is infeasible. In spite of these challenges, there are a few successful attempts at their application to systems with some form of dependencies. Levitin, for instance, in Refs. [76] and [77], respectively applied the UGF approach to systems with lateral dependencies and systems prone to CCF. Both instances, however, involved only one system with a single commodity. Stochastic PN [92, 95] and Bayesian Networks [70] are another set of powerful computational tools for the reliability modelling of systems with dependencies. However, they also require state enumeration when applied to multi-state systems, which may be infeasible for some complex system architectures.

2.3 Maintenance Modelling of Complex Systems

Owing to the rapid growth of the human population and the proliferation of new electrical technologies, the demand for sustainable electricity is on a steady rise. Coupled with a competitive market, the electrical power operator is under increasing pressure to deliver an adequate, safe, affordable, and uninterrupted supply. However, they are constrained by the impossibility of continuously operating the system without outages, because of component failures and maintenance. To minimize the impact of these outages on consumer satisfaction, the maintenance strategy adopted should be robust, meet operator expectations, extend the life of the system, and be carefully executed [33,119]. From an operator perspective, a robust strategy is one that ensures maximum system throughput and keeps the operating cost to a minimum.

In addition to its impact on system performance, maintenance accounts for a significant proportion of the total operating cost of power systems. To a significant extent, defines the revenue generated and the overall investment sustainability. In summary, the principles of modern maintenance engineering do not only require meeting technical and operational goals, but achieving them through the most cost-effective means. This constraint dictates that maintenance follow a strategy imposing minimum system output loss with the least possible cost. It is true even for other systems, as illustrated by Andrews et al., for an unmanned well-head [5] and a railway track [5], Lansey et al. [71], for a water distribution network, and Van der Duyn et al. [35], for a production system.

2.3.1 Maintenance Strategy Optimization

Maintenance can be optimized against various reliability and performance indices. The indices used depend on the application and the goal of the analyst. For instance, in nuclear and other safety-critical systems, failure probability and recovery likelihood are the most frequently used indices. However, regardless of the application and the indices used, the goal is, finding the optimum balance between costs and benefits, whilst not ignoring any important system constraints [33]. This process involves comparing the monetary equivalent of the benefits, to the costs incurred in their attainment. Consequently, cost minimization has been the subject of many maintenance optimization models [14,58,74,82,91,96,97,119,131,142,154]. A limiting factor, therefore, would be the convertibility to monetary gains, of these benefits.

While some optimization models consider the system as a single unit (for instance [94,96,119]), many are enhanced for multi-component systems. With respect to implementation effort, multi-component models are more demanding, due to the presence of multiple system dynamics and structural complexities. Nevertheless, various researchers have successfully implemented maintenance optimization models on multi-component systems [14,58,74,82,91,97,154]. A comprehensive review can be found in [10,59,99].

The cost of maintaining a system involves various parameters, varying according to the external dynamics surrounding the system and the intrinsic properties of its building block. Prominent amongst these are the reliability and maintainability of the components, cost of spares, labour cost, and the frequency and duration of preventive maintenance actions. An accurate model, therefore, accounts for all of these parameters. With a few exceptions focusing on reliability-centered maintenance [58, 154] or maintenance contract assessment [90], most models are dedicated to determining either the optimal preventive maintenance schedule, inspection, or component replacement intervals. Often, they are hinged on the assumption that there are sufficient maintenance teams to accomplish maintenance functions, [14,58,74,90,91,96,142,154], and delays imposed by logistic and administrative constraints are usually ignored [14,58,74,90,91,96,97,142,154]. Instantaneous preventive maintenance or inspection is another assumption frequently invoked [14, 74, 91, 94, 97]. While these assumptions are reasonable for some systems, they may be completely unrealistic for many. A notable instance being, a system with large maintenance durations that is maintained by limited maintenance teams. These large durations, normally due to logistic or human resource constraints, affect system performance negatively. They also render the cost and number of spares used worth considering, a factor many maintenance optimization models have ignored.

When the possibility of maintenance interruptions exists, constraints on the states of components during periods of maintenance suspension become important. A component's maintenance is suspended if it requires spares whose availability is delayed or if the maintenance team is reassigned to a more critical component. During suspensions, the component may either be put back into operation (assuming it is only partially failed or under preventive maintenance) or kept out of operation until maintenance is completed. The careful scheduling of these maintenance actions may also mitigate their effect on throughput losses. This is the case, especially for planned preventive maintenance and corrective maintenance of partially failed components. Hence, there is the need for an optimization framework that derives the combination of procedures (maintenance strategy) minimizing system losses, as well as the maintenance cost. Maintenance strategy here refers to a set of procedures specifying the following.

1. The number of maintenance teams employed and how they are assigned.
2. Whether or not maintenance should be carried out by the same team.
3. Whether the state of the system or a relevant subsystem should be taken into consideration in deciding when to commence the preventive or corrective maintenance of partially failed components.
4. What happens to a component when its maintenance is suspended.

Significant strides have already been made toward maintenance optimization in the presence of some of these, including other dynamics like ageing, imperfect, and condition-

based maintenance [9, 14, 97, 98]. However, the techniques proposed in these works are suited only to binary-state systems. An approach considering all the constraints in question and in a multi-state, multi-component environment, has yet to emerge.

2.4 Nuclear Power Plant Safety

Nuclear power is produced by harnessing the heat generated from a fission reaction chain, in a reactor vessel. The reactor vessel is placed in a concrete containment to shield the environment from the potential release of radioactive materials. Core damage occurs when the core temperature exceeds a certain threshold or the nuclear fuel elements in the vessel are uncovered. This event may trigger containment breach, inflicting huge environmental and economic catastrophe.

Severe accident mitigation is achieved in part, by ensuring reliable cooling water circulation in the reactor vessel. This objective, during normal plant operation, is achieved through heat exchange between the primary and secondary loops of the plant's main cooling system. The process, however, ceases on plant shut-down and backup cooling systems are required to sustain decay heat removal. Like the main cooling system, the backup cooling systems rely on Alternating Current (AC) provided by sources outside the plant (offsite power). When these sources fail (loss of offsite power, LOOP), emergency sources on-site are started, to drive the plant's safety systems. If the emergency sources are also unavailable or unable to function as required, the plant is said to be in a Station Blackout (SBO). The backup cooling systems, however, are equipped with alternative turbine or diesel-driven pumps to help the plant cope with SBO. These systems, on the downside, require monitoring and control via Direct Current (DC) from DC power banks. Their sustainability, therefore, regardless of inherent reliability, is limited by the DC battery depletion time. This time, and the boil-off rate of reactor coolant, define the maximum tolerable AC power recovery duration [43].

SBO accidents are the largest contributor to nuclear power plant risk, accounting for over 70% of the core damage frequency at some plants [43, 44]. LOOP events, which initiate these accidents, are classified on the basis of their origin. A grid-centred LOOP is due to the failure of the transmission network outside the plant, switchyard-centred LOOP arises from failures in the switchyard on the plant premises, plant-centred LOOP is triggered by the operational dynamics of the plant itself, while weather-related LOOP is attributed to failures induced by severe and extreme weather [43, 44]. The effective SBO risk is the sum of the core damage frequencies induced by the various LOOP types.

2.4.1 Station Blackout Risk Quantification

SBO risk quantification starts with LOOP event tree analysis [25], where the Emergency Power System availability is checked in the first heading. This event failure, whose frequency is the SBO frequency, transfers the analysis to the SBO event tree [43]. In the latter, the successes of the various mitigating actions, including offsite power and the recovery of the emergency diesel generators at specific times are also queried. These times, however, vary across plants and depend on the status of a plant's mitigating systems. At the Maanshan nuclear power plant in Taiwan, for instance, power recovery is queried at 1, 2, 4, and 10 hours into SBO. Each top event probability in the SBO event tree requires one or more static fault trees [26, 122, 139] for its quantification.

Static fault tree analysis employs an analytical approach, as such, it carries the important advantage of being computationally efficient. For this reason, its sensitivity, importance, and uncertainty analysis capabilities are outstanding, relative to simulation techniques. These attributes explain its wide use for risk analysis in the nuclear, aviation [140], and chemical process industries [66]. Unfortunately, however, the static nature of fault trees limits their applicability in many ways. For instance;

1. Implementing certain types of interdependencies is either tedious or completely impossible.

2. The analyst has to assume SBO is coincident with LOOP and that all power recovery efforts start simultaneously **after** SBO sets in. As a consequence:

   (a) the SBO frequency and non-recovery probability are overestimated in most cases, since the repair of a failed element is normally initiated immediately.

   (b) for plants with multiple emergency power systems, it is impossible to determine which sequence of response minimises the SBO frequency and maximises the recovery probability simultaneously.

   (c) it is also difficult to investigate the effects of external factors like logistical problems, extreme environmental events, and human resource constraints on the recovery process.

3. The analyst is forced to assume the non-occurrence of a second SBO after power recovery. This assumption, however, loses its validity if the emergency sources are recovered first. In this case, a second failure could initiate another SBO sequence before offsite power recovery.

4. Finally, there is the problem of inconvenience due to repetitive modelling. Since the non-recovery probability is normally required for multiple instances, each would require a dedicated fault tree.

There are numerous instances of remarkable attempts at extending the applicability of fault trees to systems with interdependencies and various forms of dynamic interactions [122, 127]. Kaiser et al. [64], for instance, introduced a state/event fault tree approach that translates fault-trees to Deterministic & Stochastic Petri Nets. Similarly, Zhou et al. [157], quite recently proposed an approach that converts static fault trees to Dynamic Uncertain Causality Graphs in order to tackle the dynamic and uncertainty attributes of practical engineering systems. However, like Kaiser's approach [64], Zhou's [157] is restricted to binary-state components and systems. Even though the performance of most components could be partitioned into two levels, the existence of multiple failure modes makes binary-state models inadequate. Also, from a modelling perspective, there are occasions when the analyst would need to model a binary-state element as a multi-state one in order to fully define its behaviour. Such flexibility requires a framework supporting multi-state modelling. Bobbio's fault tree to Bayesian Network mapping procedure [12] effectively solves this problem. However, like Kaiser's and Zhou's approaches, Bobbio's mapping procedure is also susceptible to deficiencies (3) and (4) outlined above.

Dynamic Fault Trees (DFT) [28,39,41,115] are perhaps the closest researchers have come to solving the limitations of static fault trees. Various approaches have been proposed for their solution but Markov analysis [39, 41, 101] remains the most popular. Markov modelling, however, like static fault tree analyis, becomes intractable with large systems and is only applicable to exponentially distributed transitions. Nevertheless, state explosion is no longer an issue, with the introduction of intuitive DFT software [40, 60]. Even with these developments, most of the DFT solution approaches are susceptible to deficiencies (3) and (4) outlined above. These deficiencies can only be addressed by approaches offering the flexibility to replicate the exact behaviour of the system. Such an approach, however, was put forward by Rao et al. [115], which they used to model the power supply system of a nuclear power plant. The approach simulates a system's DFT and addresses most of the limitations of static fault trees. However, like the majority of system reliability models, Rao's work is only applicable to binary-state components. The development of a more universal simulation framework, therefore, is desirable.

2.5 The Concept of Survival Signature

The operating status of an M-component system at time t can be deduced from its state vector, $\underline{\boldsymbol{x}} = (x_1, x_2, ..., x_M)$, where x_i is the state of the i^{th} component at that time. For binary-state systems, $x_i = 1$, if the i^{th} component is working and 0, if failed. Consequently, $\underline{\boldsymbol{x}} \in \{0,1\}^M$ and $x_s \in \{0,1\}$, where x_s is the state of the system.

By considering all the possible state vectors of a system, a function that maps the states of its components to its states can be obtained. This mapping, otherwise known as the structure function, $\varphi(\underline{\boldsymbol{x}})$, of the system, is an expression taking the

value 1 when the system works, and 0, otherwise. It is an algebraic representation of the system's topology, and dissociates the connectivity of its components from their probabilistic attributes. Given its structure function alone, the reliability of a system can be computed directly from the reliabilities of its components.

The structure function also finds use in system indexing and comparison [124]. However, being an algebraic expression, the possibility of multiple equivalent expressions for the same system exists. This is the case especially for topologically complex systems, which was why Samaniego [124] proposed an alternative representation of the system structure. The new representation, which he called the system signature, is an M-dimensional probability vector whose i^{th} element denotes the probability of the i^{th} component failure leading to system failure. It was hinged on the assumption that all the components of the system are identical, with independently distributed failure times. This assumption, however, is unrealistic in two ways; first, most practical systems are composed of a variety of components. Second, as discussed in Section 2.2, interdependencies exist in most systems, rendering their component failures correlated.

In response, Coolen et al. [30] proposed a new formalism, the system survival signature, to generalise Samaniego's system signature. With the survival signature, the assumption of identical components is no longer mandatory but they still must fail independently. The survival signature, $\boldsymbol{S}_\tau\left(l_1, l_2, ..., l_K\right)$, of a system with K different types of components, is the probability that the system will work when l_1 components of type 1, l_2 components of type 2, l_3 components of type 3, and so on, are working.

2.5.1 Theoretical Basics

Consider a system with K component types, with M_k components of type $k \in \{1, 2, ..., K\}$, such that $\sum_{k=1}^{K} M_k = M$. Let the random failure times of components of the same type be identical and independently distributed. Consequently, components of the same type can be grouped and defined by the set $\boldsymbol{\rho}^{\{k\}}$. Each $\boldsymbol{\rho}^{\{k\}} \forall k \in \{1, 2, ..., K\}$ is considered an independent subsystem, which gives rise to a total of K subsystems, at the system level. The system state vector can then be written as, $\underline{\boldsymbol{x}} = \{\underline{\boldsymbol{x}}_1, \underline{\boldsymbol{x}}_2, ..., \underline{\boldsymbol{x}}_K\}$, where $\underline{\boldsymbol{x}}_k$ is the state vector for subsystem k (type k components).

Now, let's modify $\underline{\boldsymbol{x}}$ to denote the actual number of available components of each component type, at a given instance. The modified system state vector, $\underline{\boldsymbol{x}}'$, is a K-element vector, such that $\underline{\boldsymbol{x}}' = \{x'_1, x'_2, ..., x'_K\}$, where x'_k, the number of available type k components, is equivalent to $\sum \underline{\boldsymbol{x}}_k$. Since components of the same type are identical, there are $\binom{M_k}{x'_k}$ state vectors, $\underline{\boldsymbol{x}}_k$, where exactly x'_k of the M_k components are working. Therefore, there are $\prod_{k=1}^{K} \binom{M_k}{x'_k}$ system state vectors, $\underline{\boldsymbol{x}}$, corresponding to $\underline{\boldsymbol{x}}'$. If this set of vectors is denoted by $\underline{\boldsymbol{X}}$, following from the definition of the survival signature (see

Section 2.5) and the fact that all the state vectors in $\underline{\boldsymbol{X}}$ are equally likely,

$$\boldsymbol{S}_{\tau}\left(\underline{\boldsymbol{x}}'\right) = \left[\prod_{k=1}^{K}\binom{M_k}{x'_k}\right]^{-1} \times \sum_{\underline{\boldsymbol{x}}\in\underline{\boldsymbol{X}}} \varphi\left(\underline{\boldsymbol{x}}\right) \tag{2.1}$$

where $\boldsymbol{S}_{\tau}\left(\underline{\boldsymbol{x}}'\right)$ is the survival signature of the system, given $\underline{\boldsymbol{x}}'$.

Let $F_k\left(t\right)$ be the common cumulative failure time distribution for all type k components, then the probability of exactly x'_k components being in operation and $M_K - x'_k$, failed, is deduced from the binomial theory as, $\binom{M_k}{x'_k}\left[F_k\left(t\right)\right]^{M_K - x'_k}\left[1 - F_k\left(t\right)\right]^{x'_k}$. Hence, the occurrence probability, $P\left(\underline{\boldsymbol{x}}'\right)$, of the state vector, $\underline{\boldsymbol{x}}'$, is expressed as,

$$P\left(\underline{\boldsymbol{x}}'\right) = \prod_{k=1}^{K}\binom{M_k}{x'_k}\left[F_k\left(t\right)\right]^{M_K - x'_k}\left[1 - F_k\left(t\right)\right]^{x'_k} \tag{2.2}$$

Therefore, the expected survival function of the system, given $\underline{\boldsymbol{x}}'$, is the product, $\boldsymbol{S}_{\tau}\left(\underline{\boldsymbol{x}}'\right) \times P\left(\underline{\boldsymbol{x}}'\right)$. The survival function or the reliability, $R\left(t\right)$, of the system, is the sum of the expected survival functions yielded by all its modified state vectors. For a system with K component types, there are $\prod_{k=1}^{K}\left(M_k + 1\right)$ such state vectors and,

$$R\left(t\right) = \sum_{\underline{\boldsymbol{x}}'\in\underline{\boldsymbol{X}}'}\left[\boldsymbol{S}_{\tau}\left(\underline{\boldsymbol{x}}'\right) \times P\left(\underline{\boldsymbol{x}}'\right)\right] \tag{2.3}$$

where $\underline{\boldsymbol{X}}'$ is the global set containing all the modified state vectors, $\underline{\boldsymbol{x}}'$, of the system.

2.5.2 Applications and Limitations

Equation 2.3 segregates the structure function of the system from the probabilistic properties of its components, and exemplifies the key advantage of survival signatures in reliability analysis. Since a system's structure function changes only with changes in its topology, survival signature-based approaches are a computationally efficient alternative for maintenance optimization, uncertainty, and sensitivity analysis problems, where only the probabilistic attributes of the system change. For these problems, the survival signature of the system is computed just once and reused in multiple reliability analyses. Other techniques, however, would require the evaluation, directly or otherwise, of both the probabilistic and topological attributes of the system, on every reliability analysis.

Since its introduction, the survival signature has been invoked in various ways, gradually gaining popularity in system reliability analysis. Aslett et al. [7], for instance, incorporated it into their Bayesian framework for system reliability analysis. Reed [118] used BDD to develop an efficient and exact algorithm to compute system survival signatures. Feng et al. [49] went a step further by proposing an analytical approach for analysing systems with imprecision in component failure time distributions. Patelli et al. [111], on the other hand, proposed a generic simulation approach for computing

the reliability of complex systems, using the survival signature. However, these works assume full independence between the component failure times. In fact, only Coolen, Eryilmaz, and Coolen-Maturi have made realistic attempts at extending the notion of survival signature to systems with interdependencies. In 2014, Coolen and Coolen-Maturi proposed a predictive model [31] that deduces the number of components that fail, as well as the subsequent reliability of the system, following a CCF event. Their model, however, stops short at computing the overall effect of CCF on the system and is applicable only to systems with a single component type. Most importantly, it does not consider the second type of induced failure, cascading failures. Eryilmaz, Coolen, and Coolen-Maturi later adapted the survival signature to compute the importance measures [46] and mean residual life [47] of coherent systems with dependent components. Their adaptations were based on the theory of weak exchangeability [45] of component failure times. With this theory, components of the same type can be dependent and have exchangeable failure times while components of different types may or may not be dependent. Its only downside, however, is its need for knowledge of the joint survival function of the components, prior to system analysis. While this is not impossible, it is in no way straight-forward for complex systems with nested cascade failures.

The survival signature is also inapplicable to multi-output systems with competing demand, as well as multi-state systems, suggesting that it has yet to be made applicable to most practical systems.

2.6 Chapter Summary

Complexities are an inevitability in practical engineering systems. A system can be deemed complex from the point of view of its topology, the interdependencies between its components, or the size of its state-space. The latter, strictly speaking, is not an element of complexity but an attribute rendering the reliability analysis of the system tedious. Hence, reliability analysts have always perceived it an element of complexity.

In this chapter, an overview of complex systems, the basis of their classification, and a review of the available computational tools for their reliability modelling, have been presented. These tools can either be analytical or simulation-based. Analytical techniques exhibit a superior computational efficiency over their simulation counterparts. However, they suffer a setback when solicited for realistic systems. Often, the reliability analyst is not only interested in the steady-state or instantaneous reliability measures but also the underlying probability distributions governing the behaviour of the system. The effects of external factors (e.g., restrictive maintenance schemes, human, environmental and other stochastic external factors) on the values assumed by these measures are another area of keen interest. When faced with such requirements, simulation is the most feasible alternative. In summary, both analytical and simulation procedures are restricted by their various unique limitations. The development, therefore, of a

framework combining some or all of the desirable attributes of these procedures would be very valuable.

Failure is inherent in all mechanical and electronic components, either as a consequence of their material properties or as a consequence of their operating conditions. To minimize the frequency and duration of failures in a system, the operator adopts preventive and corrective maintenance, respectively. There is, therefore, no way a system can operate without scheduled and unscheduled interruptions. In a multi-component system, maintenance activities have to be scheduled and executed with prudence, to satisfy both performance and cost constraints. The planning of maintenance to satisfy system performance and cost constraints is maintenance optimization. An overview of this process, with a detailed review of the available techniques, has also been discussed.

Finally, the chapter presents an overview of the basic operating principles of a nuclear power plant, placing emphasis on station blackout accidents, the largest contributor to its risk. Station blackout accidents are initiated by the loss of the external AC power supply to a plant. They occur on the complete failure of the plant's standby power systems before the restoration of the external power supply. Their risk is computed via a static fault tree analysis which applicability is inhibited by its inability to intuitively model induced interdependencies, as well as complex maintenance strategies and other operational dynamics in systems. The chapter has presented a detailed review of the applicability of fault trees and other risk modelling frameworks, to station blackout risk modelling and quantification.

Chapter 3

Multi-state System Reliability Modelling & Evaluation

3.1 Introduction

In chapter 2, we learned that, the structural complexity of systems, coupled with their multi-state characteristics, renders their reliability evaluation difficult. Notwithstanding the emergence of various techniques dedicated to multi-state system analysis, simulation remains the only approach applicable to realistic systems. However, most simulation algorithms are either system-specific or limited to simple systems when solicited for multi-state system reliability analysis. This is due to their inability to compute the actual flow corresponding to a given system configuration. Therefore, they require the enumeration of all the possible system states, defining the cut sets associated with each state, and monitoring their occurrence. In addition to being extremely tedious for large and complex systems, state enumeration and cut-set definition require a detailed understanding of the system's failure mechanism. In this chapter, a simple and generally applicable simulation approach, enhanced for multi-state systems of any topology, is presented. In the approach, each component is defined as a semi-Markov stochastic process and via discrete-event simulation, the operation of the system is mimicked. The principles of flow conservation are invoked to determine the flow across the system for every performance level change of its components, using the interior-point algorithm [68, 100]. This eliminates the need for cut set definition and overcomes the limitations of existing techniques. The methodology can be exploited to account for the effects of transmission efficiency and loading restrictions of components on system reliability and performance. The principles and algorithms developed, and which are published in [55], are applied to two numerical examples, to demonstrate their applicability.

The remainder of this chapter is organised into five sections. A detailed overview of the proposed approach and its advantages over existing techniques are presented in Section 3.2. Section 3.3 describes the modelling procedure for components, followed

in Section 3.4, by a description of how the system is modelled. The latter section also contains the details of the simulation procedure, as well as its associated limitations. The numerical case studies are presented in Section 3.5, which also analyses the simulation's computational expenses. Finally, a summary of the chapter constitute Section 3.6.

3.2 Overview of Proposed Approach

The approach is based on the fact that if the properties of the components of a system are known, its performance can be deduced from its network model. Therefore, each component is modelled as a multi-state object with a defined state-space and property set for each state. The operation of the system is simulated by initially sampling the next state and transition time (hereafter referred to as transition parameters) of each component. The component with the earliest transition time is identified and its sampled next state made its current state. At this stage, the system state vector (vector of the current capacities of all the components of the system) is updated and used to compute the output of the system via a linear programming algorithm. Using its new state, the next transition parameters of the component whose transition has just been forced are sampled and the component that will make a transition next identified. This cycle of sampling the transition parameters of the components, forcing the required transitions, and computing the output of the system continues until the mission time is just exceeded. As the simulation progresses, the system performance computed at every component transition is captured and saved. From this saved history, the reliability and performance indices of the system are computed at the end of the simulation. To replicate the actual operating principles of most practical systems, a special component shut-down and restart procedure is incorporated. In this procedure, the availability of each system component is tested against its predefined reference minimum input load level at every transition. With this, the effects of functional interdependence (see Section 2.2) on the failure probability of the components are accounted for.

A common feature exhibited by the components of most realistic systems is transmission inefficiency, a term ascribed to the phenomenon where an intermediate component transmits only a fraction of the flow received from its preceding component. The component, in other words, acts as a partial sink and dissipates part of the flow, such that the flow transmitted to the next component is less than that received. Under this condition, flow is no longer conserved, and techniques built around the flow conservation principle become obsolete. To illustrate the effect of transmission inefficiency on system reliability, consider a 50MW power generator supplying a 45MW load through a 75MW transformer. If there are no power losses in the transformer, 45MW will be transmitted to the load. However, if its efficiency declines to 75%, it now takes all 50MW from the generator but transmits only 37.5MW. In both cases, the apparent difference between capacity and demand remains constant but the power drawn from the generator

increases and the effective power supplied to the load declines. Other examples are a power transmission line prone to losses and an oil pipeline in which a mode of failure could be a hole in a pipe or gasket failure at some flange.

The transmission inefficiency of the components of a system most often exert undesirable effects on its performance and reliability. It is, therefore, worthwhile considering this effect in its reliability evaluation process. In the scenario discussed in the preceding paragraph, the generator would need to be rated at least 60MW to match demand. Therefore, if the system is not equipped with adequate controls, such that the capacity constraint is always satisfied, the generator may fail due to overloading, even though the demand remains well below its rated capacity. Considering this in the proposed approach, each component state is assigned a performance level and a sink index, respectively defining the maximum flow accepted or generated by the component and the proportion dissipated when residing in that state.

The convenient representation of a system's architecture and evaluating its output from the state changes of its components are the two prime difficulties encountered in the simulation of complex multi-state systems. These difficulties are overcome in the proposed approach by using adjacency matrices invoked from network theory to define the structure of the system. Adjacency matrices are a square array of 1's and 0's depicting the connectivity of a network and can easily represent any system architecture. They can be manipulated to obtain system flow equations, which are solved to determine the magnitude of flow through every node of the system. Their efficiency stems from their elimination of the need to define the cut sets of the system or enumerate its states, prior to analysis. Complex network theory is a widely used concept and finds application in many engineering and real-life problems. It is particularly useful in representing and analysing complexities in system structure. As a result, many researchers have applied its principles to a variety of problems, yielding excellent results. For instance, Dwivedi et al. [42] used it in the vulnerability analysis of a power system, Todinov [132] established his optimization of repairable flow networks entirely on it, and Chen et al. [29] used it to analyse the reliability/availability of the Manhattan street networks.

3.2.1 Advantages Over Existing Techniques

The following list highlights the key contributions of the proposed technique for multi-state system reliability evaluation.

1. Being simulation based, it inherits all the advantages of simulation approaches to system reliability and performance evaluation. With respect to other simulation algorithms, it can implement any system structure with relative ease, since it doesn't require knowledge of the minimal path or cut sets prior to system analysis.

2. It is not maximum-flow-based. Instead, it calculates the actual flow across every node of the system. This, consequently, makes possible the following:

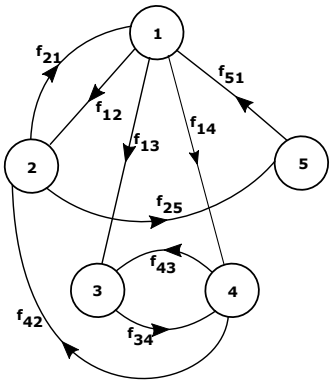


Figure 3.1: State-space diagram of a particular multi-state component.

(a) analysis of systems with multiple source and sink nodes with competing demand which can be static or instantaneous;

(b) analysis of systems prone to losses at nodes and across edges;

(c) the easy restart and shut-down of components, where necessary, so as to replicate the actual operation of systems.

3. Node capacities and system demand can take on any positive value. They do not necessarily have to be integer-valued, as required by other graph-based algorithms.

3.3 Component Modelling

Multi-state components and systems exist in only one state at a given instant but reside continuously in that state until a transition occurs. The transition instantaneously takes them to another state where they reside until the next transition [11]. These transitions are defined by time-dependent probability distributions or some stochastic event outside the component boundary and whose underlying probability distribution may or may not be known. The interactions between the states of an n-state component and their corresponding transition probability density functions can be represented by an n-order square transition matrix, $\boldsymbol{T}$. As an example, Figure 3.1 shows the state-space diagram of a certain five-state component, where the label beside each arc represents the probability density function of the time-to-occurrence of the transition depicted. If the transition from state x to y is represented by the pair, (x, y), and defined by its probability density function, $f_{xy}(t)$, each element at position (x, y) of $\boldsymbol{T}$ is equal to $f_{xy}(t)$. Assuming all the possible transitions from state x have the same priority, the next state, y_{next}, of the component depends only on which transition occurs first.

Occasionally, some component transitions are controlled by the direct effect of events outside the component boundary. For instance, in a series connected system, the failure of one component, necessitates the shut-down of the components in operation. The

shut-down components are restarted only after the failed component is restored. In such a system, the shut-down and restart of a component is triggered by the failure and repair of another component. However, the incidence and duration of these failure and repair events are marked by uncertainty, which means it is impossible to assign a time-to-occurrence probability distribution to the transitions they induce. Also, the induced transitions may not be Markovian, in that, the component's next state can also depend on its previous state(s). These transitions are regarded as forced transitions throughout this thesis, as they are induced by events outside the component boundary. The other transitions, defined by known probability distributions and independent of events outside the component boundary, are regarded as normal transitions. Hence, $\boldsymbol{T}$ defines the stochastic behaviour of the component, as outlined in Equation 3.1.

$$\begin{aligned} \boldsymbol{T} &= \{f_{xy}(t)\}_{n\times n} \mid x \neq y \quad (x,y) \in \{1,2,...,n\} \\ \boldsymbol{T}(x,y) &= \begin{cases} \infty, & \text{If (x,y) is a forced transition} \\ 0, & \text{If no transition between x \& y} \\ f_{xy}(t), & \text{Otherwise} \end{cases} \end{aligned} \tag{3.1}$$

The capacity (performance), c_x, of a binary-state component in state x is such that, $c_x \in \{0, c_{max}\}$, where $c_x = c_{max}$, if the component is working, and 0, otherwise. The capacity space of a multi-state component, on the other hand, is defined by the set $0 \leqslant c_x \leqslant c_{max}$, c_{max} being the maximum capacity of the component. Each component state has an associated capacity that defines the maximum flow the component can generate, transmit, or sink in that state. The performance, therefore, of a multi-state component can be defined by the vector, $\boldsymbol{C}$, of state capacities, as in Equation 3.2.

$$\boldsymbol{C} = \{c_x\}^n \mid 0 \leqslant c_x \leqslant c_{max}, \quad n \geqslant 2 \tag{3.2}$$

$$\boldsymbol{\varsigma} = \{\varepsilon_x\}^n \mid 0 \leqslant \varepsilon_x \leqslant 1 \tag{3.3}$$

Equation 3.3 defines the vector of sink indices of a multi-state component. The sink index, ε_x, is the amount of flow dissipated in the component when in state x, as a fraction of the total flow it receives. Applicable only to intermediate components (components that are neither sources nor sinks/targets), this property ranges between 0 and 1. A value of 0 means outflow is equal to inflow while a value of 1 corresponds to the case when all the flow is dissipated in the component. With a sink index of 1, a component effectively becomes a sink, explaining the choice of name. By default, sources have a sink index of 0 and sinks, a sink index of 1.

Sometimes, a component may be subjected to loading restrictions, in order to preserve its reliability and/or ensure its safe operation. This often requires the load to exceed a threshold value but lie within the maximum load rating of the component.

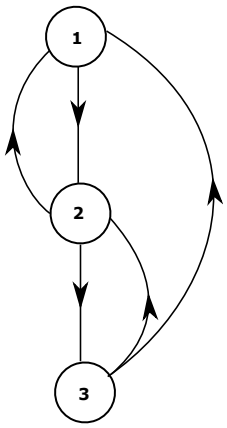


Figure 3.2: State-space diagram of 40MVA generator.

Outside this range, the component is either shut down or considered failed. The maximum load rating corresponds to the maximum capacity of the component but the threshold load rating, however, may be greater than its minimum capacity. This is so because the component may be prone to complete failures or maintenance states characterised by 0 capacity. The minimum load rating Λ is such that $0 \leqslant \Lambda \leqslant c_{max}$.

$$\mathbb{E}_i = (\boldsymbol{T}, \boldsymbol{C}, \boldsymbol{\varsigma}, \Lambda, x_0), \tag{3.4}$$

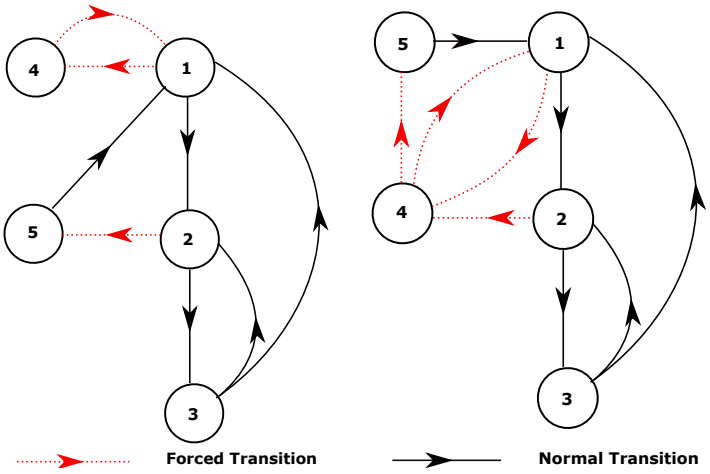
Each parameter discussed determines a component's behaviour and subsequently, its effect on system performance and reliability. Therefore, with x_0 as its initial state, a component can be defined by the quintuple $\mathbb{E}_i$ as expressed in Equation 3.4.

3.3.1 Application to Repairable Multi-state Components and Systems

A simple illustration will be used to describe the modelling of a multi-state component with repairable partial failure modes. Consider a 40MVA generator subjected to a 20% minimum load restriction and existing in three possible states as follows:

1. state 1, depicting operation at its nominal output level;
2. state 2, depicting operation below its nominal level due to partial failure;
3. state 3, depicting its total failure.

Shown in Figure 3.2 is a representation of the interactions between the states of the generator. State 2 is the state of interest in this illustration and the objective is to explore the possible events embedded in the restoration process from partial failure (that is, transition $2 \rightarrow 1$). The figure is based on the assumption that the generator remains in operation whilst undergoing repairs from state 2 (on-line repairs). However, this is not the case for components of many real-world engineering systems. Most components would need to be taken out of operation before repair actions can be completed, which



(a) Immediate repair (b) Repair only after shut-down

Figure 3.3: Alternative state representation of generator.

event may trigger the unavailability of other components, as explained in Section 3.3. The assumption, therefore, may result in over/under estimation of certain reliability indices, when wrongly applied.

Assume the generator is in series with an external breaker which can fail and undergo off-line repairs (that is, the breaker needs to be taken out of operation during maintenance). During the off-line repair of the breaker, the generator too would need to be shut down. To account for this induced unavailability of the generator, a new state, 4, is introduced into the state-space of the generator, as shown in Figure 3.3.

The restoration of the generator from state 2 to 1 may follow one of two possible operational pathways:

1. Case 1: Repair is initiated as soon as the generator enters state 2. This, technically means, the generator does not exist in states 2 and 3.

2. Case 2: Repair is delayed until the generator is not in operation. That is, either when totally failed or when shut down from partial failure.

These considerations necessitate the introduction of a fifth state, 5, to account for the period when the generator is undergoing repairs from state 2 to 1. The state-spaces of the generator for each of Case 1 and Case 2 are presented in Figure 3.3. In the figure, forced and normal transitions are differentiated. Forced transitions, however, would have to be manually effected during simulation, as they may depend on the state history of the component. For instance, the generator goes to state 5, from state 4, if its previous state is 2, else, it goes to state 1, as shown in Figure 3.3(b). With reference to Case 2, the parameters of the generator are as defined in Equation 3.5. The sink index for each state is 0, since the generator is a source. As a rule of thumb, the sink

index is not required for sources and sinks, as will soon be evident.

$$\boldsymbol{T} = \begin{pmatrix} 0 & f_{12}(t) & 0 & \infty & 0 \\ 0 & 0 & f_{23}(t) & \infty & 0 \\ f_{31}(t) & f_{32}(t) & 0 & 0 & 0 \\ \infty & 0 & 0 & 0 & \infty \\ f_{51}(t) & 0 & 0 & 0 & 0 \end{pmatrix} \tag{3.5}$$

$$\boldsymbol{C} = (40 \quad 25 \quad 0 \quad 0 \quad 0)$$

$$\Lambda = 8$$

The modelling technique described in this section can be extended to components with multiple partial failure modes. This is achieved by introducing two states (one each, for shut-down and repair) for each partial failure state.

3.3.2 Determining Component State Transition Parameters

Algorithm 1 Sampling procedure for transition parameters of a multi-state component.

Require: x and t are respectively the component's current state and simulation time

function SAMPLE(x,$\boldsymbol{T}$,t)
  $\boldsymbol{J}$ ←set of possible transitions from state x
  $\boldsymbol{f}$ ←set of corresponding distributions
  k ←Number of elements in $\boldsymbol{J}$
  for n ←1 to k **do** ▷ Loop over possible transitions
    $(\boldsymbol{t}_{times}, n) \leftarrow (\boldsymbol{f}, n)$ Ⓢ1 ▷ Generate 1 sample from the n^{th} element of $\boldsymbol{f}$
  end for
  $t_{sample} \leftarrow min\,(\boldsymbol{t}_{times})$ ▷ get the minimum sampled time
  $\boldsymbol{p}$ ←transitions corresponding to t_{sample}
  if $numel(\boldsymbol{p}) > 1$ **then** ▷ if multiple transitions
    $u \sim [0, 1]$ ▷ generate uniform random number
    $index \leftarrow (\boldsymbol{p}, \lceil u * numel(\boldsymbol{p}) \rceil)$
  else
    $index \leftarrow \boldsymbol{p}$
  end if
  $y_{next} \leftarrow index$ ▷ get next state
  $t_{next} \leftarrow t_{sample} + t$ ▷ get next transition time
  return $(y_{next}$,$t_{next})$
end function

The key to the simulation of multi-state systems is being able to correctly determine the next states and transition times (transition parameters) of its components. The proposed algorithm for sampling the next transition parameters of a multi-state component is based on the assumption that its next state depends only on its current state. Starting with the component in state x at time t the algorithm is summarised thus.

1. Locate all non-zero elements in row x of $\boldsymbol{T}$, saving their next possible states in the set, $\boldsymbol{J}$, of possible transitions from x.

2. Define a set, $\boldsymbol{f}$, containing the distributions corresponding to the elements of set $\boldsymbol{J}$ in step 1.

3. Sample each element of $\boldsymbol{f}$ and save the sampled values in $\boldsymbol{t}_{times}$.

4. Find the minimum, t_{sample}, of $\boldsymbol{t}_{times}$ and define a set, $\boldsymbol{p}$, containing the next states corresponding to t_{sample}, such that $\boldsymbol{p} \subset \boldsymbol{J}$.

5. The next state, $y_{next} = \boldsymbol{p}$, if $\boldsymbol{p}$ has only one element. Otherwise, it is an element randomly selected from $\boldsymbol{p}$.

6. The next transition time, t_{next}, is the sum, $t_{sample} + t$.

These steps are summarised as a pseudo-code by Algorithm 1. Its major advantage is its capability to ensure an unbiased determination of the transition parameters of components including, those exhibiting both deterministic and probabilistic state transitions. The algorithm will never select a forced transition over one exhibiting Markovian properties. As a rule of thumb, it is not applied when the component resides in states from which only forced transitions are possible (e.g, state 4 in Figure 3.3). The simulation algorithm should, therefore, be equipped with special routines to force these transitions.

3.4 System Modelling

In complex network theory, system topology is defined by a graph, which is a set of nodes connected by edges or links across which some material (can be real or virtual) is transmitted. This material may be the current flowing in a circuit, the energy generated by a power plant, the liquid pumped from a reservoir, the traffic flow rate in a street network, or any quantifiable quantity of interest. In a graph, there is a set, $\boldsymbol{s}$, of nodes generating the controlled information and another set, $\boldsymbol{t}$, utilising the information. Nodes belonging to $\boldsymbol{s}$ are source nodes whilst those belonging to $\boldsymbol{t}$ are sink or target nodes. Between the sources and sinks, are nodes facilitating the transmission process, ensuring the sources and sinks remain connected. These are intermediate nodes and they, with the sources, influence the quality and success of the communication process.

If flow is in only one direction through every edge, the graph is said to be directed. Otherwise, it is undirected or bidirectional.

3.4.1 The System as a Directed Graph

As established in Chapter 2, engineering systems are designed to accomplish a specific, often quantifiable process. Process flow may be possible in any direction, depending on the connectivity of the system and the properties of its components. However, at any instance, the direction of the process is known and fixed [42]. Systems are normally a collection of components performing different but specific roles, and these components, together, determine the success of the process. Some of the components may be responsible for initiating the process whilst some are just links to ensure its progression. There is also another actor normally external to the system that utilises the process and drives it through the system. The set of components initiating the process are analogous to sources in a graph, the components serving as links, to intermediate nodes, and so is the external factor driving the process analogous to sinks. Hence, the topology of the system can be accurately represented by a directed graph.

$$G = (\boldsymbol{V}, \boldsymbol{A}) \tag{3.6}$$

Since the aim in system reliability evaluation is to investigate the effects of component failure on system performance and life-span, the structure of the system can be represented by a directed graph with components and output points considered nodes connected by perfectly reliable edges i.e., edges do not fail. If G is a directed graph, the structure of the system is defined by G, as in Equation 3.6, where $\boldsymbol{V}$ is the set of nodes and $\boldsymbol{A}$, the adjacency matrix. When modelling systems like power distribution networks, with unreliable links, each unreliable link should be treated as a component.

$$\boldsymbol{A} = \{a_{ij}\}_{M\times M} \mid a_{ij} = \begin{cases} 1 & \text{If flow is } i \rightarrow j \\ 0 & \text{Otherwise} \end{cases} \tag{3.7}$$

$$\boldsymbol{e} = \{i, j\}_{k\times 2} \mid k = \sum_{j=1}^{M}\sum_{i=1}^{M} a_{ij} \quad \forall (i,j) \in \boldsymbol{V} \tag{3.8}$$

Let the components of the system, including load/demand points be consecutively numbered from 1 to M. The adjacency matrix is an M-order square matrix defining the connectivity of nodes. A connection between nodes i and j is represented by a '1' at the intersection of row i and column j of $\boldsymbol{A}$, if process flow is from node i to j ($i \rightarrow j$) and a '0', if a connection does not exist. The node pair, (i, j), representing a connection and flow, from node i to j, is known as an edge/link, e_{ij}, of the system. The set of edges of the system is defined by a $k \times 2$ matrix, $\boldsymbol{e}$, where k, the total number of edges,

is equal to the sum of the elements of $\boldsymbol{A}$. Equations 3.7 and 3.8 respectively define the adjacency and edge matrices of the system.

$$\boldsymbol{A} = \{a_{ij}\}_{M\times M} \mid a_{ij} = \begin{cases} \alpha_{ij} & \text{If flow is } i \rightarrow j \\ 0 & \text{Otherwise} \end{cases} \tag{3.9}$$

The physical links, in some systems, may be reliable but inefficient, which condition imposes negative effects on the system's reliability and performance. Let α_{ij} be the efficiency of edge e_{ij}, such that $0 < \alpha_{ij} \leqslant 1$. Links with zero efficiency are sinks, and should, therefore, be treated as nodes. The adjacency matrix could be redefined to convey the information about the efficiency of the links, as well. Equation 3.7 could, therefore, be generalised, as expressed by Equation 3.9 and k redefined as the total number of non-zero elements of $\boldsymbol{A}$, such that, $k = \sum_{j=1}^{M}\sum_{i=1}^{M}(a_{ij} > 0)$.

$$G = (\boldsymbol{V}, \boldsymbol{A}, \boldsymbol{L}) \tag{3.10}$$

Algorithm 2 Procedure for deriving the edge and incidence matrices of a system.

Require: $\boldsymbol{A}$, the adjacency matrix of the system

function GETMATRIX($\boldsymbol{A}$)
  $M \leftarrow size(\boldsymbol{A}, 1)$ ▷ get number of nodes
  $k \leftarrow \sum_{j=1}^{M}\sum_{i=1}^{M}(a_{ij} > 0)$ ▷ get number of edges
  $\boldsymbol{e} \leftarrow \{0\}_{k\times 2}$ ▷ Predefine $\boldsymbol{e}$
  for $n \leftarrow 1$ to M **do** ▷ Loop over nodes
    $\boldsymbol{\Delta} \leftarrow$set of columns of $\boldsymbol{A}$ with non-zero entries
    $w \leftarrow numel(\boldsymbol{\Delta})$ ▷ get number of elements in $\boldsymbol{\Delta}$
    $\boldsymbol{e}\,(\Downarrow w) \leftarrow \left[n \times \{1\}_{w\times 1} \quad \boldsymbol{\Delta}^T\right]$ ▷ update next w rows of $\boldsymbol{e}$
  end for
  delete all zero elements in $\boldsymbol{e}$
  $\boldsymbol{\Gamma} \leftarrow \{0\}_{M\times k}$ ▷ predefine the incidence matrix by an M by k array of zeros
  $\boldsymbol{i} \leftarrow$vector of elements in column 1 of $\boldsymbol{e}$
  $\boldsymbol{j} \leftarrow$vector of elements in column 2 of $\boldsymbol{e}$
  $position \leftarrow \boldsymbol{j}^T + \{0, 1, ..., k-1\} \times M$
  $position1 \leftarrow \boldsymbol{i} + (\boldsymbol{j} - 1) \times M$
  $(\boldsymbol{\Gamma}, position) \leftarrow -(\boldsymbol{A}, position1)$ ▷ update incidence matrix
  $position2 \leftarrow \boldsymbol{i}^T + \{0, 1, ..., k-1\} * M$
  $(\boldsymbol{\Gamma}, position2) \leftarrow 1$ ▷ update incidence matrix
  return ($\boldsymbol{e}$,$\boldsymbol{\Gamma}$)
end function

In addition to inefficiency considerations, the links of most realistic systems, transmission lines in power distribution systems, for instance, have a maximum load they can safely transmit. Let l_{ij} denote the maximum allowable load for link e_{ij}, such that

$0 < l_{ij} \leqslant \infty$, with the upper bound of the range corresponding to the case when no load restrictions are imposed on the edge. The capacity matrix, $\boldsymbol{L}$, is introduced as a third property of the system graph model, to account for restrictions on how much load each link can safely transmit. Though unrealistic, especially for power systems,'no load restriction' is useful in maximum flow analysis of distribution networks. Like $\boldsymbol{A}$, $\boldsymbol{L} \mid \boldsymbol{L} = \{l_{ij}\}_{M\times M}$ is an M-order square matrix, with each element corresponding to an element in the former. Equation 3.6, therefore, can be adapted to define both the structure of a system and the properties of its links, as expressed by Equation 3.10.

$$\boldsymbol{\Gamma} = \{\gamma_{pq}\}_{M\times k} \mid \gamma_{pq} = \begin{cases} 1, & p = i \\ -a_{ij}, & p = j \\ 0, & \text{otherwise} \end{cases} \qquad \forall (i,j) \in \boldsymbol{e} \tag{3.11}$$

Given the adjacency, edge, and capacity matrices of a system, a fourth matrix defining the relationship between its edges and nodes, with respect to the direction of process flow, will be introduced. This matrix, referred to as the the incidence matrix, $\boldsymbol{\Gamma}$, has a dimension of $M \times k$, with its rows and columns corresponding to the nodes and edges of the system, respectively. If the edges are numbered from 1 to k_e and nodes, from 1 to M, for link e_{ij}, the element in the column of $\boldsymbol{\Gamma}$, corresponding to the edge number of the link (that is, the index of (i,j) in matrix $\boldsymbol{e}$) and row i is assigned the value 1 and the element in row j (the incident/local target node), assigned the value $-a_{ij}$. This process is repeated for all the links of the system, and the vacant locations in $\boldsymbol{\Gamma}$ are each filled with a '0'. Mathematically, $\boldsymbol{\Gamma}$ is defined as expressed by Equation 3.11. The variable $q = 1, 2, ..., k$ (the edge number) is the index of edge e_{ij} in $\boldsymbol{e}$ and $p = 1, 2, ..., M$. Given the adjacency matrix alone, $\boldsymbol{e}$ and $\boldsymbol{\Gamma}$ can be derived by applying Equations 3.8 and 3.11, as summarised by Algorithm 2.

3.4.2 System Representation and Flow Analysis

The transition matrix is usually unknown for the output nodes of a system, prior to its reliability analysis. Their sink indices, too, are not required, as evident in the system flow equations. For these nodes, therefore, Equation 3.4 simplifies to $\mathbb{E}_i = (\mathbf{C}, \Lambda)$.

$$\mathbb{S} = (G, \mathbb{E}) \mid \mathbb{E} = \{\mathbb{E}_i\}^M, \quad i = 1, 2, ..., M \tag{3.12}$$

If $\mathbb{E}$ is the set containing the properties, $\mathbb{E}_i$, of each node of the system, the system structure and property can be defined by the set, $\mathbb{S}$, as expressed by Equation 3.12.

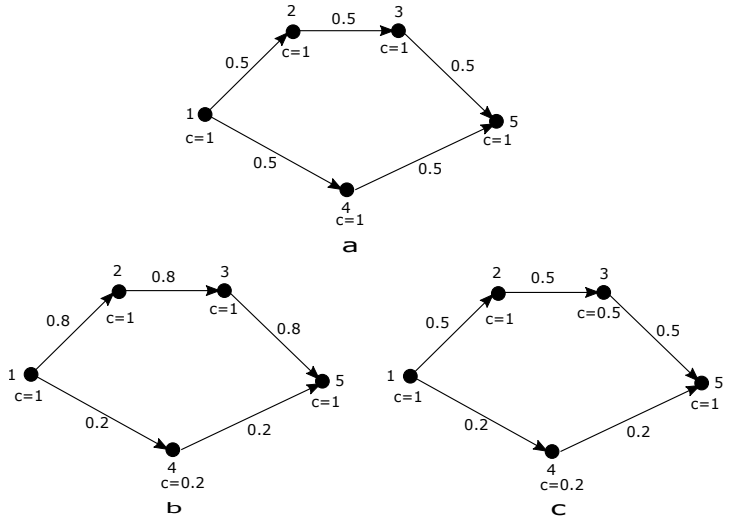


Figure 3.4: Flow visualisation in a particular 5 node system.

3.4.2.1 Derivation of System Flow Equations

To understand the flow of processes in systems, it is worthwhile thinking of sources as reservoirs and intermediate nodes as valves in a pipe network. The total flow through the system depends on the amount of liquid in the reservoir and the resistance to flow, which properties are a function of the capacities of the source and intermediate nodes, respectively. As the source and intermediate nodes perform their functions, their capacities change, leading to changes in the resistance of some flow paths. The result is the existence of paths with high resistance and some with a low resistance, triggering differential flow redirection. The amount of flow through a path is directly proportional to the capacity of the smallest node in that path. Hence, more flow will be redirected through the path of least resistance (highest capacity), as illustrated in Figure 3.4.

In Figure 3.4, node 1 is a source, node 5, a sink, and nodes 2, 3 & 4 are intermediate nodes. Initially (see Figure 3.4(a)), all the intermediate nodes have the same capacity, resulting in equal path resistance for the upper and lower paths between source and sink. Therefore, the 1 unit of flow from the source is split equally between them, as shown. In Figure 3.4(b), the capacity of node 4 reduces to 0.2, thereby creating a difference in the resistances of the two paths. Since the maximum capacity of the upper path is 1, 0.8 units of flow are redirected through it and 0.2 through the lower path. If at this stage, the capacity of node 3 is also reduced to 0.5 (see Figure 3.4(c)), the effective capacity of the upper path reduces to 0.5. This explains why only 0.5 units are transmitted along the upper path and 0.2 along the lower path. It is, however, worthwhile noting that while the flows in Figures 3.4(a) and (b) are valid, they are only a subset of the possible flow combinations that can be transmitted between nodes 1 and 5, without violating the flow conservation and capacity constraints of the system. In Figure 3.4(a), for instance, all the 1 unit of flow could be transmitted along any of the two paths,

and nothing along the other. Similarly, in Figure 3.4(b), all the 1 unit of flow could be transmitted along the upper path, and nothing along the lower path. Regardless of the flow combination, however, the total flow at the output node is always the same. Flow redirection may be simple and straight-forward for simple systems but presents itself as an optimization problem for systems with complex topology. The problem consists of multiple parameters and constraints, and involves calculating the flow along each link of the network. Even when the full capacity of the sources can be transmitted across the network, it is impossible to determine, without resorting to linear programming, the magnitude of flow into each output node, save for the case when every node operates at its maximum capacity.

Let X_{ij} be the magnitude of flow in link e_{ij}, $\mho_i^+$; the set of nodes connected to the inlet of node i, $\mho_i^-$; the set of nodes connected to its outlet, and $c_x^{\{i\}}$; its current capacity. The total inflow for sources is zero and for sinks, the total outflow is zero. This implies, $i \in \boldsymbol{s}$, if $\mho_i^+ = \varnothing$ and $i \in \boldsymbol{t}$, if $\mho_i^- = \varnothing$. The constraints, and by extension, the equations governing the optimization procedure, hinge on the following assumptions.

1. The system is equipped with adequate controls, such that flow does not exceed the capacity of a node. This means, for intermediate and sink nodes, the total inflow should not exceed the node capacity. This is known as the capacity constraint, and it is expressed for intermediate and sink nodes as,

$$\sum_{j \in \mho_i^+} X_{ji}\alpha_{ji} \leqslant c_x^{\{i\}} \mid (i,j) \in \boldsymbol{e}, \quad \mho_i^+ \subset \mathbf{V} \tag{3.13}$$

   For sources, the statement implies the total outflow should not exceed the node capacity. Hence,

$$\sum_{j \in \mho_i^-} X_{ij} \leqslant c_x^{\{i\}} \mid (i,j) \in \boldsymbol{e}, \quad \mho_i^- \subset \mathbf{V} \tag{3.14}$$

   By applying Equation 3.13 to all intermediate and sink nodes and Equation 3.14, to all sources, a single equation, Equation 3.15, expressing the capacity constraint on the system can be obtained. $\boldsymbol{\Theta}$ will be called the inequality constraint matrix.

$$\boldsymbol{\Theta}\{X_{ij}\}_{k\times 1} \leqslant \{c_x^{\{i\}}\}_{M\times 1} \mid (i,j) \in \boldsymbol{e}, \quad \forall i \in \mathbf{V} \tag{3.15}$$

$$\boldsymbol{\Theta} = \{\theta_{iq}\}_{M\times k} \mid \theta_{iq} = \begin{cases} \gamma_{iq}, & i \in \boldsymbol{s} \\ -\gamma_{iq}, & \gamma_{iq} < 0 \\ 0, & \text{otherwise} \end{cases} \tag{3.16}$$

   $\boldsymbol{\Theta}$ is related to the incidence matrix, $\boldsymbol{\Gamma}$, of the system by Equation 3.16.

2. Flow in the system is conserved. That is,

- The total flow generated by sources equals the sum of flow consumed by sinks and any losses at intermediate nodes.
- The total inflow at an intermediate node, $i \mid i \in (\boldsymbol{V} - (\boldsymbol{s} \cup \boldsymbol{t}))$, equals its total outflow plus any losses at the node.

$$\sum_{j \in \mho_i^-} X_{ij} - \sum_{j \in \mho_i^+} X_{ji}\alpha_{ji} = 0 \mid (i,j) \in \boldsymbol{e} \tag{3.17}$$

$$\boldsymbol{\Phi}\{X_{ij}\}_{k\times 1} = \{0\}_{\eth \times 1} \quad \forall (i,j) \in \boldsymbol{e} \tag{3.18}$$

$$\boldsymbol{\Gamma} = \begin{pmatrix} \Gamma_1 \\ \Gamma_2 \\ . \\ . \\ . \\ \Gamma_{M-1} \\ \Gamma_M \end{pmatrix} = \{\Gamma_p\}^M \mid p = 1, 2, ..., M \tag{3.19}$$

$$\boldsymbol{\Phi} = \begin{pmatrix} \Phi_1 \\ \Phi_2 \\ . \\ . \\ . \\ \Phi_{\eth-1} \\ \Phi_{\eth} \end{pmatrix} = \{\Phi_\lambda\}^{\eth} \mid \Phi_\lambda = \Gamma_p, \quad \lambda = 1, 2, ..., \eth \tag{3.20}$$

$$\eth < M, \quad f : \lambda \to p \quad \forall p \in (\boldsymbol{V} - (\boldsymbol{s} \cup \boldsymbol{t}))$$

Expressing the second statement mathematically, Equation 3.17 is obtained. By applying this equation to all the intermediate nodes of the system, the flow conservation constraint equation, Equation 3.18, is obtained. $\eth$ is the number of intermediate nodes and $\boldsymbol{\Phi}$ will be called the equality constraint matrix.

If the incidence matrix, $\boldsymbol{\Gamma}$, is expressed in terms of its rows, Γ_p, as in Equation 3.19, $\boldsymbol{\Phi}$ and $\boldsymbol{\Gamma}$ are related as expressed in Equation 3.20. By arranging the intermediate nodes in ascending order of their ID, Equation 3.20 suggests, the λ^{th} row of $\boldsymbol{\Phi}$ is identical to the p^{th} row of $\boldsymbol{\Gamma}$, where p is the λ^{th} element of the ordered set of intermediate nodes. $\boldsymbol{\Phi}$, therefore, is a sub matrix of $\boldsymbol{\Gamma}$, containing all the rows of the latter corresponding to intermediate nodes.

$$\sum_{j \in \mho_i^-} X_{ij} - (1 - \varepsilon_x^{\{i\}}) \sum_{j \in \mho_i^-} X_{ji}\alpha_{ji} = 0 \mid (i,j) \in \boldsymbol{e} \tag{3.21}$$

$$\begin{aligned}
&\boldsymbol{\Phi} = \{\phi_{\lambda q}\}_{\eth \times k} \mid \phi_{\lambda q} = \begin{cases} (1 - \varepsilon_x^{\{p\}})\gamma_{pq}, & \gamma_{pq} < 0 \\ \gamma_{pq}, & \text{otherwise} \end{cases} \\
&\lambda = 1, 2, ..., \eth, \quad \eth < M \\
&f : \lambda \rightarrow p \quad \forall p \in (\boldsymbol{V} - (\boldsymbol{s} \cup \boldsymbol{t}))
\end{aligned} \tag{3.22}$$

However, a close look at Equation 3.17 would reveal that the inefficiency of the intermediate nodes has been ignored, in its formulation. When this is considered, Equation 3.17 is rewritten as in Equation 3.21, where $\varepsilon_x^{\{i\}}$ is the sink index of node i. With this consideration, the equality matrix, $\boldsymbol{\Phi}$, is generalised by Equation 3.22.

$$\begin{aligned}
\mathbb{O} &= -\sum_{j \in \mho_i^-} \sum_{i \in s} X_{ij} \\
&= -\{\psi_q\}_{1 \times k} \{X_{ij}\}_{k \times 1} \mid \psi_q = \sum_{i \in s} \gamma_{iq} \\
q &= 1, 2, ..., k
\end{aligned} \tag{3.23}$$

3. If the capacity of a node changes, the flow in the system is reconfigured to match the prevailing system conditions. The flow generated by sources is such that the total flow into the sinks is maximised. The objective of the optimization problem, therefore, is to determine the minimum source flow that maximises the system flow. Hence, the objective function, $\mathbb{O}$, is as defined in Equation 3.23.

4. The minimum flow through link e_{ij} is 0 but its maximum flow, Ω_{ij}, is defined by the capacities of its nodes, as well as its own capacity, l_{ij}. That is, $0 \leqslant X_{ij} \leqslant \Omega_{ij}$. If $\boldsymbol{lb}$ and $\boldsymbol{ub}$ are the vectors respectively containing the minimum and maximum flows across each link of the system, then,

$$\begin{aligned}
\boldsymbol{lb} = \{0\}_{k \times 1}, \quad \boldsymbol{ub} = \{\Omega_{ij}\}_{k \times 1} \\
\Omega_{ij} = \min\{c_{max}^{\{i\}}, c_{max}^{\{j\}}, l_{ij}\} \quad \forall (i,j) \in \boldsymbol{e}
\end{aligned} \tag{3.24}$$

Equations 3.15, 3.18, 3.23, and 3.24 form the basis of the optimization procedure, which can be implemented by a variety of well-known algorithms. However, the numerical examples presented in this thesis are based on the interior-point algorithm [68, 100]. Of the parameters required, only $\boldsymbol{\Phi}$ and the set, $\{c_x^{\{i\}}\}_{M \times 1}$, of current node capacities, vary during simulation. In fact, even $\boldsymbol{\Phi}$ does not vary for systems with no losses at the nodes.

$$\Omega_{gh} = 0 \mid (g,h) = \begin{cases} (j,i) & \text{If } X_{ij} > X_{ji} \\ (i,j) & \text{Otherwise} \end{cases} \tag{3.25}$$

So far, only unidirectional links have been considered. Without loss of generality, the equations proposed in this section, are valid for bidirectional links, as well. Such links are normally represented by two reciprocal edges i.e., edges connecting the same

pair of nodes but allowing flow in opposing directions. For instance, edges e_{12} and e_{21} are reciprocals. The optimization procedure sometimes results in flow across both edges. However, since in practice the edges represent the same physical link and flow at any instance is in only one direction (see Section 3.4.1), the calculated values should be normalised, to obtain the effective flow through the link. The effective flow direction is the same as the direction of the larger flow while its magnitude is obtained by temporarily setting the maximum capacity of the edge with the smaller flow to zero and repeating the optimization procedure, as expressed by Equation 3.25.

Following the termination of the linear programming algorithm, the vector, $\boldsymbol{\eta}$, of flows through the nodes of the system is given by $\boldsymbol{\Theta}_{M\times k}\{X_{ij}\}_{k\times 1}$.

3.4.2.2 Output Calculation and Node Reconfiguration

In a system, the failure and repair of one component can impose a corresponding change in the outputs of other components. The change, therefore, may trigger the shut-down of operating components and the restart of components in shut-down. It is imperative that these shut-down and restart events are accounted for in the simulation of the system, as the failure of most components depends on the time spent in operation.

Shutting down a component affects all other components connected to it and hence, process flow in the system. Therefore, the effective output of a node after the transition of another node, is the flow through it after the last component is shut-down. For this, an iterative procedure is employed to calculate the system output. The procedure is based on the following assumptions and principles.

1. The availability of a node is determined by the magnitude of its input flow only.
2. A node which current capacity is non-zero is shut-down when the flow through it is zero or below its predefined threshold.
3. Nodes are shut-down in descending order of their rank.
4. Nodes with zero input flow are placed higher on the scale.
5. Non-zero input flow nodes are ranked in order of their degree of inadequacy (i.e percentage by which the input falls short of the threshold)
6. Equally ranked nodes have the same priority and are shut-down randomly.

Highlighted below is the iterative procedure proposed for system output calculation.

1. Calculate the system flow, using Equations 3.15, 3.18, 3.23, and 3.24.
2. Find the nodes requiring shut-down and define a set, $\boldsymbol{U_0}$, to hold the ones with zero-input and a set, $\boldsymbol{U}_+$, for the rest. Go to step 7 if $\boldsymbol{U_0}$ and $\boldsymbol{U}_+$ are empty.

3. Shut down all the nodes in $\boldsymbol{U_0}$.

4. For each node shut down,

   (a) save its next transition parameters, set its current capacity to zero, its next transition time to infinity, and add the node to the set, $\boldsymbol{\delta}$.

   (b) add it to the maintenance list if a transition to a maintenance state from shut-down is possible and force maintenance.

   (c) remove the node from its set.

5. Rank the nodes in $\boldsymbol{U}_+$, shut down the top ranked node, and call step 4.

6. Repeat steps 1 to 5.

7. Determine the magnitude of flow through the output node.

When nodes are shut down, a complementary restart operation is carried out to restore them to their previous states. Enumerated below are the steps entailed.

1. Provided $\boldsymbol{\delta} \neq \varnothing$, calculate the system flow using equations 3.15, 3.18, 3.23, and 3.24 with the sink indices and capacities, before shut-down, of the nodes in $\boldsymbol{\delta}$.

2. Select a node in shut-down.

3. Check if its input flow satisfies its threshold requirements.

4. If it does, determine the time, t_{spent}, it spent in shut-down.

5. Restore the node to its previous state and update its next failure time. The new failure time, t'_{next}, of the node becomes $t_{next} + t_{spent}$.

6. Remove the node from the maintenance list if it is repairable only in shut-down.

7. Remove the node from $\boldsymbol{\delta}$ and repeat steps 2 to 6 until $\boldsymbol{\delta} = \varnothing$.

8. End procedure

3.4.3 Simulation Procedure

Summarised below are the systematic steps proposed for the simulation of structurally static homogeneous time-dependent systems. Homogeneous systems, as used in this thesis, are those transmitting only one type of flow/material.

1. Initialise the system, in preparation for simulation. This involves the following,

   (a) initialization of the vectors, $\{c_x^{\{i\}}\}_{M\times 1}$; to save the current capacities, $\{\varepsilon_x^{\{i\}}\}_{M\times 1}$; to save the current sink indices, and $\boldsymbol{\tau} \mid \boldsymbol{\tau} = \{\infty\}_{M\times 1}$; to save the next transition times of nodes. Also initialise the vectors to save the performance and state history (where necessary) of all the nodes of the system.

(b) setting the required number of simulations, N, and mission time, T_m.

2. Set the simulation time, $t = 0$, the set of nodes in shut-down, $\boldsymbol{\delta} = \varnothing$, sample the next transition parameters of source and intermediate nodes, and update $\boldsymbol{\tau}$.

3. Compute the flow across the system and save the flow through each node.

4. Shut down nodes, where necessary, as described in Section 3.4.2.2.

5. Set the current simulation time to the minimum of $\boldsymbol{\tau}$. That is, $t = min\,(\boldsymbol{\tau})$.

6. Check for nodes with $t_{next} = t$, and for each,

   (a) effect the required transition and sample its next transition parameters.

   (b) update $\boldsymbol{\tau}$, $\{c_x^{\{i\}}\}_{M\times 1}$, $\{\varepsilon_x^{\{i\}}\}_{M\times 1}$ and its state and performance history.

   (c) add to the maintenance list if the new state is a maintenance state or if transition to a maintenance state from this new state is possible.

7. For each node on the maintenance list, force maintenance.

8. Compare the previous and current values of $\{c_x^{\{i\}}\}_{M\times 1}$ and $\{\varepsilon_x^{\{i\}}\}_{M\times 1}$. If a difference is observed in at least one of the two vectors,

   (a) restart any nodes in shut-down

   (b) determine the new value of matrix $\boldsymbol{\Phi}$

   (c) calculate system flow, using the new values of $\boldsymbol{\Phi}$, $\{c_x^{\{i\}}\}_{M\times 1}$, and $\{\varepsilon_x^{\{i\}}\}_{M\times 1}$ and shut down nodes, where necessary. Note that all other parameters of Equations 3.15, 3.18, 3.23 and 3.24 remain static throughout the simulation

   (d) update the flow/performance history of nodes for which the current flow differs the flow at the previous system transition.

9. Repeat steps 5 to 8 until $t = T_m$, updating $\boldsymbol{\tau}$, $\{c_x^{\{i\}}\}_{M\times 1}$, $\{\varepsilon_x^{\{i\}}\}_{M\times 1}$, as well as the state and flow histories of nodes, at every transition.

10. Repeat steps 2 to 9 N times, saving the node histories at each trial.

Given the state and performance histories, the desired reliability and performance indices can be obtained using standard statistical analyses. These analysis, however, fall outside the scope of this chapter but details can be found in [56, 89], as well as Chapter 4 of this thesis. Indices such as failure distribution, failure frequency, reliability function, average availability, instantaneous availability, instantaneous output, state probabilities, capacity factor, meant-time-to-fail, mean-time-between-failures, and the actual distributions of forced transitions, are obtainable from the simulation histories.

The proposed state duration-based simulation technique achieves superior accuracy, relative to the standard sequential Monte Carlo simulation described in [125], when

applied to repairable systems. The upper hand is due to the shut-down and restart of nodes, as a consequence of the failure and repair of other nodes. The standard technique, on the other hand, assumes the nodes are statistically independent. It should be stressed, however, that this shut-down and restart procedure reduces the computational speed of the proposed technique. The proposed procedure may also require both the state and flow changes of all the nodes to be saved, as the simulation progresses. This increases its computational burden and reduces its speed. In practice, the state histories of all the nodes is required only for maintenance optimization problems. For basic system reliability and availability analysis, the flow history of sink/output nodes only, is required. The user, therefore, must use their discretion in deciding which aspect of the algorithm to adopt and which to modify, based on the problem being solved.

3.4.4 Limitations of the Proposed Approach

The proposed simulation procedure is challenged by two major limitations. The first is consequent of the assumptions used for shutting down and restarting nodes. In the approach, the availability of a node is determined by the magnitude of its input flow only, restricting its applicability to homogeneous and independent heterogeneous systems. However, it can be easily extended to interdependent systems by incorporating fault trees (or a more flexible tool) and redefining the conditions for shut-down and restart.

The second limitation is due to the capacity constraint imposed on source and intermediate nodes. This implies the effects of cascading failures resulting from flow redistribution cannot be studied. However, the approach can be used in system design to estimate the required system parameter values to prevent these failures.

3.5 Case Studies

The principles and algorithms derived and described in the preceding sections were translated into a Matlab-based application and applied to two case-studies. The random variable generator available in the open-source uncertainty quantification tool, OpenCossan [110], developed at the Institute for Risk and Uncertainty of the University of Liverpool, was incorporated to enhance sampling from any probability distribution.

3.5.1 Case Study 1: A Simple Pipe Network

Consider the 3-component pipeline shown in Figure 3.5, adapted from [89]. A maximum of 4 tons of oil could be pumped from the source, X_{in}, to the output, X_{out}, where the demand is fixed at 3.5 tons. The state-space of each of the other components is shown, with the number beside each state denoting the capacity of the component in that state. Beside each arc is the transition rate (in transitions per year) of the transition depicted.

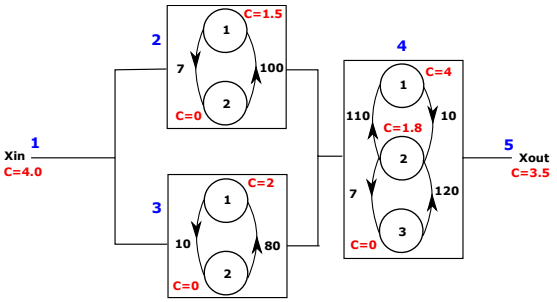


Figure 3.5: A 3-component pipe network.

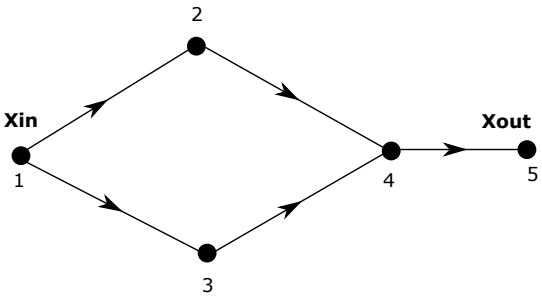


Figure 3.6: Network model of pipe network.

The problem under review was initially solved by Lisnianski et al. [89] using Markov Chain, whilst assuming on-line repairs to node 4. The system, in this case study, is analysed, also, for this scenario and an additional two scenarios, as summarised thus.

1. Assuming on-line repairs to node 4 (the scenario originally analysed).
2. Node 4 is taken out of operation during repairs, which repairs commence almost instantaneously as the node enters a degraded state.
3. Node 4 is taken out of operation during repairs, which repairs only commence after the component is shut down due to the failure of another node.

3.5.1.1 Analyses

The equivalent graph model of the system is shown in Figure 3.6. Notice the two extra nodes, 1 and 5, representing the source and sink, respectively. Assuming the efficiency and capacity of the links are of no interest in the analysis, they can be assumed to be 100% efficient. Similarly, if there are no losses in the system, the vector of sink indices is defined as, $\{0\}_{M\times 1}$. The available information is sufficient to formulate the linear programming problem and derive its parameters. The first step in this is to define the adjacency matrix, since all the other parameters depend on it. From Figure 3.6, the adjacency, edge, and incidence matrices are as expressed by Equations 3.26 and 3.27.

$$\boldsymbol{A} = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{pmatrix} \tag{3.26}$$

$$\mathbf{e} = \begin{pmatrix} 1 & 2 \\ 1 & 3 \\ 2 & 4 \\ 3 & 4 \\ 4 & 5 \end{pmatrix} \quad \mathbf{\Gamma} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & -1 & 1 \\ 0 & 0 & 0 & 0 & -1 \end{pmatrix} \tag{3.27}$$

$$\begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X_{12} \\ X_{13} \\ X_{24} \\ X_{34} \\ X_{45} \end{pmatrix} \leqslant \begin{pmatrix} 4.0 \\ 1.5 \\ 2 \\ 4 \\ 3.5 \end{pmatrix} \tag{3.28}$$

$$\begin{pmatrix} -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} X_{12} \\ X_{13} \\ X_{24} \\ X_{34} \\ X_{45} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \tag{3.29}$$

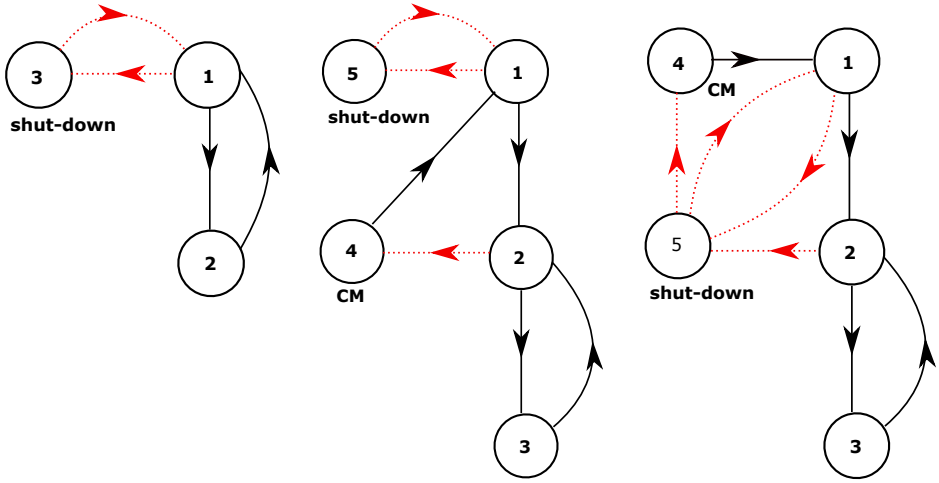
$$\boldsymbol{lb} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \qquad \boldsymbol{ub} = \begin{pmatrix} 1.5 \\ 2 \\ 1.5 \\ 2 \\ 3.5 \end{pmatrix} \tag{3.30}$$

$$\mathbb{O} = \begin{pmatrix} -1 & -1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} X_{12} \\ X_{13} \\ X_{24} \\ X_{34} \\ X_{45} \end{pmatrix} \tag{3.31}$$

$$\mathbf{\Theta} = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \tag{3.32}$$

With $\boldsymbol{A}$, $\boldsymbol{e}$, and $\mathbf{\Gamma}$ known, the linear programming problem is formulated as follows.

1. At time 0, all the components are in their best states. The inequality and equality constraints, therefore, are expressed as in Equations 3.28 and 3.29, respectively.
2. The bounds on the flows through the edges are defined by Equation 3.30.
3. Finally, the objective function is as expressed in Equation 3.31.



(a) Nodes 2 & 3 in all 3 cases (b) Node 4 in Case 2 (c) Node 4 in Case 3

Figure 3.7: State-space diagram of components.

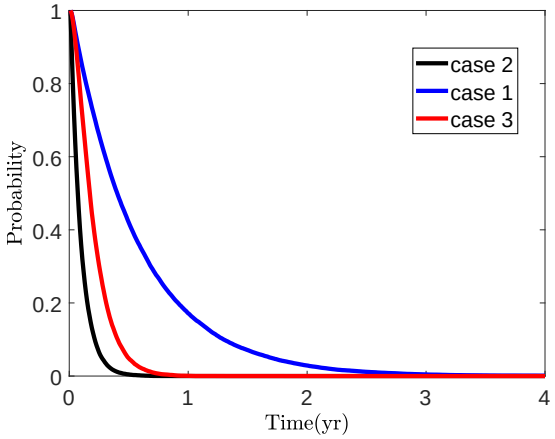


Figure 3.8: Reliability function.

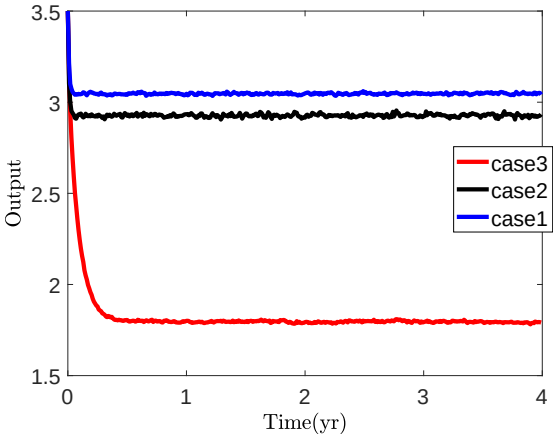


Figure 3.9: System instantaneous output.

By comparing Equations 3.28 and 3.15, Equation 3.32 can be deduced. Similarly, comparing Equations 3.29 and 3.18 would reveal Equation 3.33.

$$\mathbf{\Phi} = \begin{pmatrix} -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & -1 & -1 & 1 \end{pmatrix} \tag{3.33}$$

With the relevant parameters, the system was analysed and the results compared, to deduce the effects of the various assumptions on its reliability and performance. The multi-state models of its nodes are illustrated in Figure 3.7, where state 'CM' is a corrective maintenance state. The models are based on the proposition in Section 3.3.1.

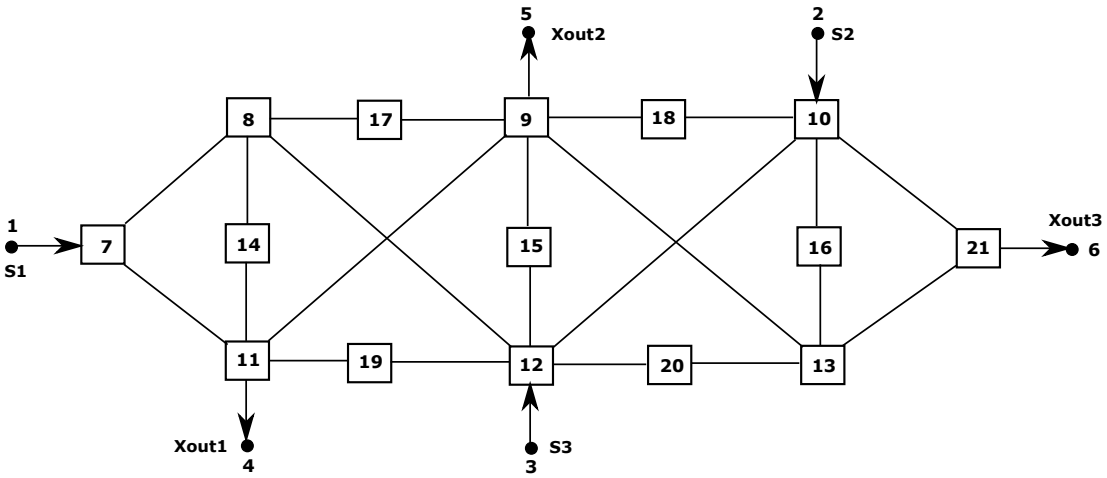


Figure 3.10: Block diagram of test bridge network.

3.5.1.2 Results and Comments

For a mission time of 4 years and 2×10^4 simulation samples, the reliability and performance indices of the system obtained from the simulation are presented in Figures 3.8 and 3.9. As shown, modelling node 4 according to case 1 would result in over estimation of the performance indices, if the system were operated according to case 2 or 3.

Though the problem analysed is simple, it illustrates the effects of modelling error on the accurate estimation of a system's reliability and performance indices.

3.5.2 Case Study 2: A Multi-State Bridge Network

Bridge networks are a typical example of structural complexity exhibited by engineering systems. The reliability analysis of even the simplest bridge network is cumbersome, when compared to the analysis of a similarly sized simple system.

To illustrate the applicability of the proposed approach to complex systems, the arbitrary multi-output and non-repairable bridge network shown in Figure 3.10, is considered. In the network, nodes 1, 2, and 3, respectively designated S1, S2, and S3 are sources while nodes 4, 5, and 6 are sinks labelled Xout1, Xout2, and Xout3, respectively. The sources have identical failure characteristics (note that failure times are in hours) but the capacity of S1 is 1.5 times the individual capacities of the other sources. Also, the demand at the sinks is fixed but that at node 6 (70 units) is twice those at nodes 4 and 5. Each source exists in three states, Working (W), Partial Failure (PF), and complete failure (F) while the intermediate nodes are either working or completely failed. In addition, all the diagonal links in the network are unidirectional, with flow from left to right. The intermediate nodes, according to their position in the system and similarity in failure characteristics, are grouped and arbitrarily designated as follows.

1. Nodes 7 to 13 and node 21, referred to as central nodes.

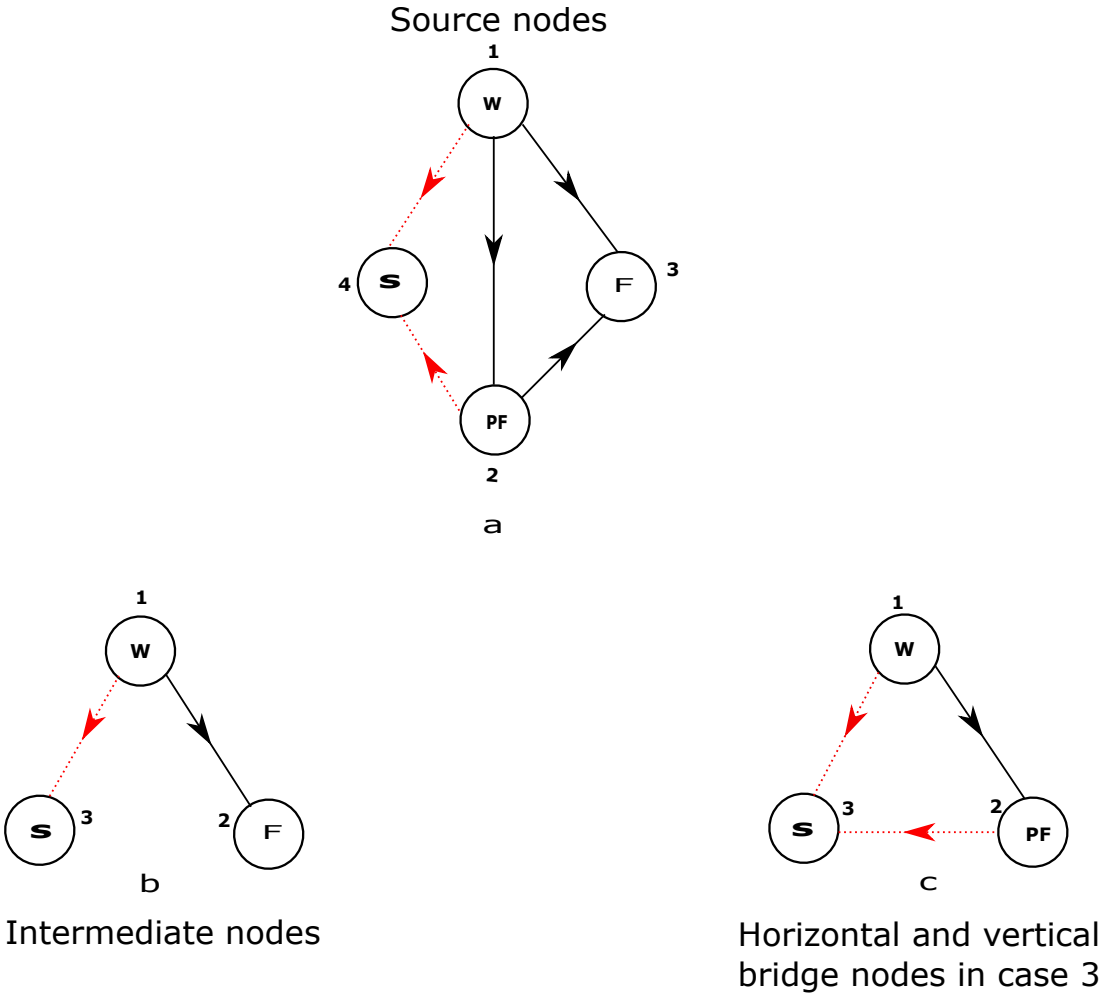


Figure 3.11: State-space diagram of system nodes

2. Nodes 14 to 16, referred to as vertical bridge nodes.
3. Nodes 17 to 20, referred to as horizontal bridge nodes.

3.5.2.1 Analyses

The network was analysed for three cases, as summarised thus.

1. The vertical bridge nodes are bidirectional but flow through the horizontal bridge nodes is from left to right.
2. Both the vertical and horizontal bridge nodes are bidirectional.
3. Both the vertical and horizontal bridge nodes are bidirectional and their complete failure state is replaced with a failure state that has the same capacity as their working state but with a reduced efficiency of 80%.

Table 3.1: System node properties.

Node Type	Transition	Distribution	Capacity
	1-2	Exp(10)	
S1	1-3	LogN(20, 2)	(60 45 0 0)
	2-3	LogN(4.5, 1.2)	
Central Nodes	1-2	LogN(15, 3)	(70 0 0)
Bridge Nodes	1-2	Wb(12, 2)	(35 0 0)

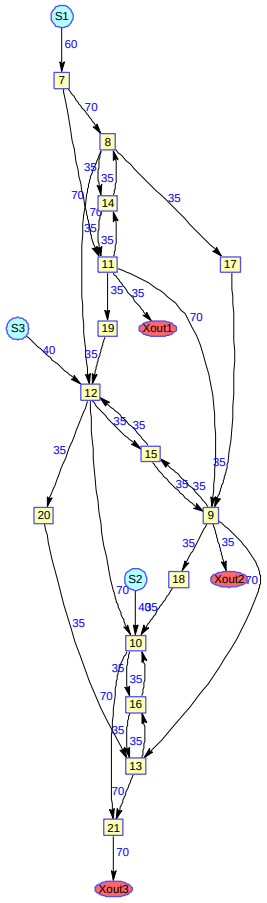


Figure 3.12: Graph for case 1.

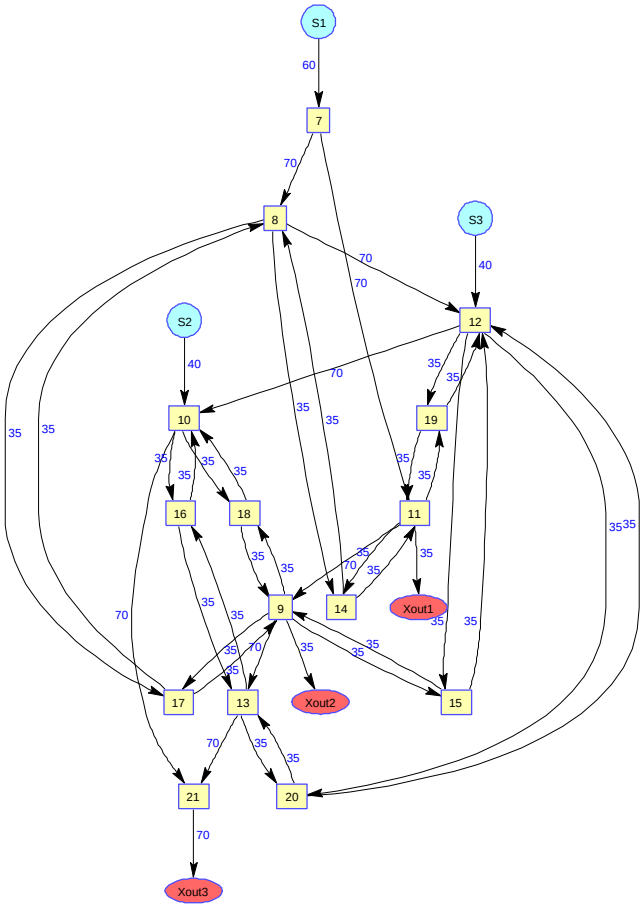


Figure 3.13: Graph model for cases 2 & 3.

Table 3.2: Reliability indices for Xout1.

Indices	Case 1	Case 2	Case 3
	0.1072	0.1074	0.1070
State Probability	0.4040	0.4683	0.5276
	0.4887	0.4244	0.3655
Capacity Factor	0.4345	0.4433	0.4059
Availability	0.5113	0.5756	0.6345
MTTF	10.2306	11.5172	12.6959
No. of Failures	0.9996	0.9996	0.9995

Figure 3.11 shows the state-space representations of the system nodes, which properties are presented in Table 3.1. State 'S' is a shut-down state, introduced to denote

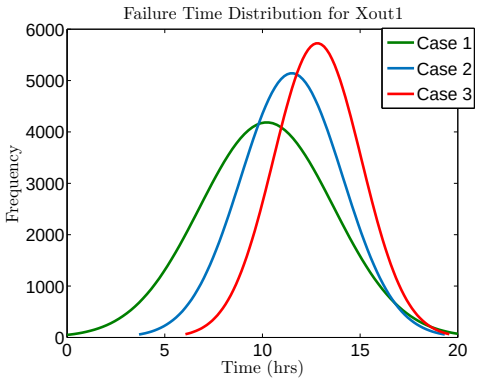


Figure 3.14: Failure time dist. at Xout1.

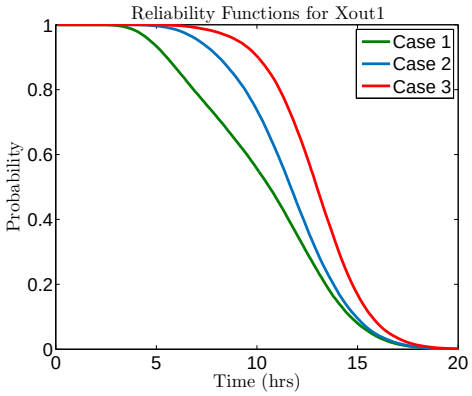


Figure 3.15: System reliability at Xout1.

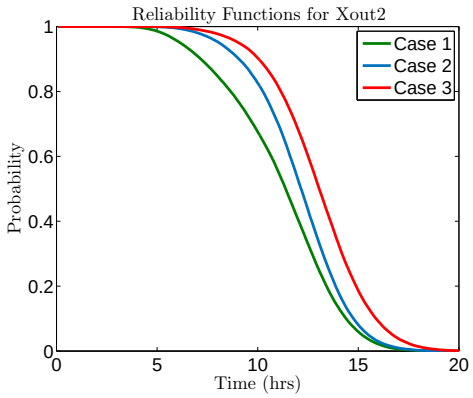


Figure 3.16: System reliability at Xout2.

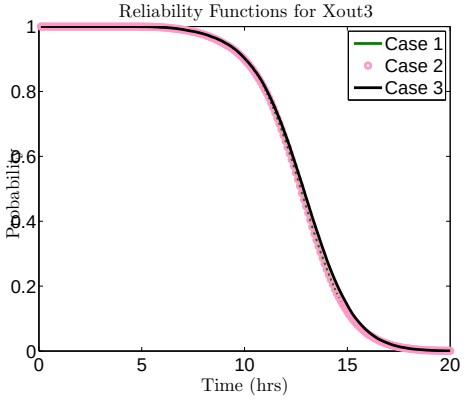
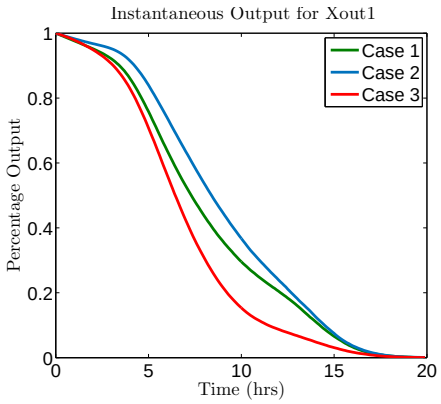


Figure 3.17: System reliability at Xout3.

the period a node is out of operation as a consequence of the failure of another node. Sources are modelled as shown in Figure 3.11(a) and the central nodes, as in Figure 3.11(b), in all three scenarios. The vertical and horizontal bridge nodes, however, are modelled as in Figure 3.11(b) in cases 1 and 2, and Figure 3.11(c), in case 3. Their capacity and sink index vectors are respectively, (35 35 0) and (0 0.2 0), in case 3. With this as the only exception, the sink index vector is irrelevant for this system, since there are no flow losses in the other scenarios. The properties of sources S2 and S3 have not been included in Table 3.1 because they can be deduced from those of S1.

3.5.2.2 Results and Comments

Shown in Figures 3.12 and 3.13 are the graph models of the system for all three scenarios, with an indication of the maximum allowable load through each link. The system was simulated for a mission time of 20 hours and 6×10^4 samples, in each case. Table 3.2 summarises the average values of the popular reliability and performance indices at node 4. The failure time distribution of the same node, under each of the three assumptions, is plotted in Figure 3.14. Failure (state 3) is defined as the complete absence of flow



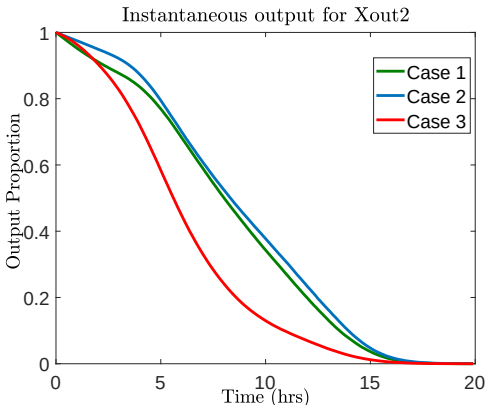
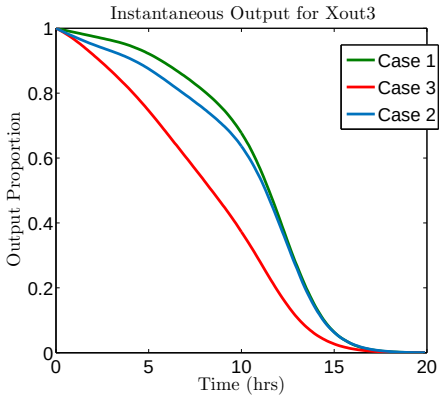


Figure 3.18: Instantaneous output at Xout1. **Figure 3.19:** Instantaneous output at Xout2.



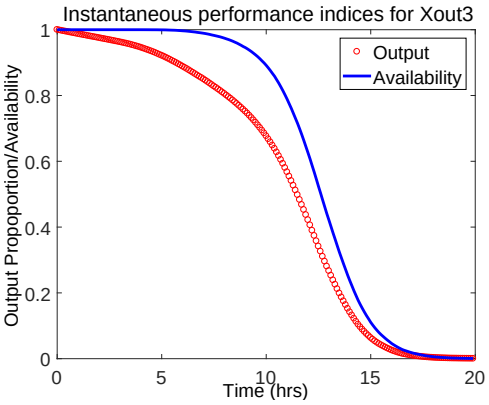


Figure 3.20: Instantaneous output at Xout3. **Figure 3.21:** Performance indices at Xout3.

at the node. The other states, states 1 and 2, denote the periods when the flow at the node matches demand and when it is below demand but non-zero, respectively. The choice of node 4 is arbitrary, as the goal is to demonstrate the possibility of computing any reliability index of interest, from the system simulation history.

The reliability at nodes 4, 5, and 6, are presented in Figures 3.15, 3.16, and 3.17, and their instantaneous outputs, in Figures 3.18, 3.19, and 3.20, respectively. System reliability, in this case study, is defined as the probability of a non-zero flow at a node. The instantaneous output of a node on the other hand, is the ratio of its flow to its capacity as a function of time. From the results, the reliability at node 6 does not vary with case, though its instantaneous output does. This observation is a consequence of the reliability function, as defined in this case study, not regarding the relative magnitude of flows. For instance, a flow of 0.4 units has the same reliability value as a flow of 30 units. Therefore, node 6 being a terminal node, system configuration and efficiency variations trigger a corresponding variation in the magnitude of flow realised at its input but not in its complete failure. Consequently, its availability is greater than its instantaneous output for about 95% of the mission, as shown in Figure 3.21.

This case study illustrates the effects of system configuration, as well as component

Table 3.3: Actual computational cost per case study.

	Case Studies	
	1	2
Average time per sample (s)	3.92	0.93
Estimated total time (s)	78400	55722
Actual simulation time (s)	3163	2128.27
Improvement factor	24.79	26.18

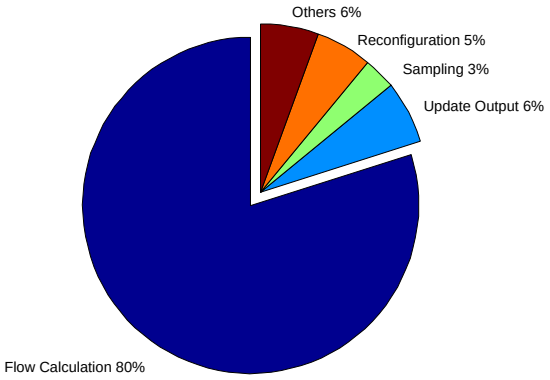


Figure 3.22: Allocation for case study 1.

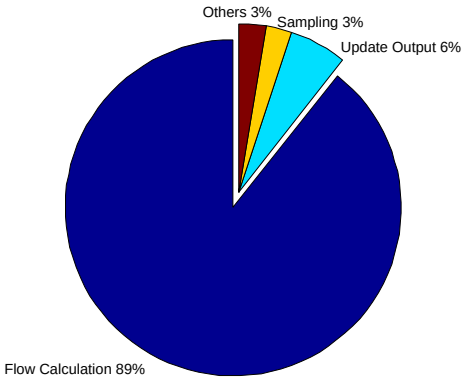


Figure 3.23: Allocation for case study 2.

efficiency, on the reliability and performance of a system. It exemplifies why reliability should not be used as the only parameter to study the response of a multi-state system to design and operational variations. It also shows that the response of an output node depends on its position in the system relative to other output nodes and sources.

3.5.3 Computational Cost of Approach

To investigate the computational requirement of each subroutine of the simulation algorithm, a series of computational experiments were performed on the systems presented in the case studies. The results, summarised by Table 3.3 with Figures 3.22 and 3.23, are based on the average of the simulation times recorded in the experiment. Table 3.3 compares what the simulation times would be if samples were run serially on a single core with the actual times spent using 24 cores running in parallel. The estimated time is the product of the simulation time per sample and the total number of samples (20000 and 60000 for case studies 1 and 2, respectively). Ideally, the improvement factor, the ratio of estimated to actual simulation time, should be slightly less than the number of parallel cores, due to initialization and overhead communication among workers. However, certain system initialization tasks which would be repeated for every sample in the serial simulation, are performed only once and broadcast across the 24 cores.

Flow calculation, as depicted by Figures 3.22 and 3.23, is the most computationally intensive subroutine. Its computational expense depends on the nature and size of the system (that is, whether or not the system is repairable), mission time, and the total

number of simulation samples. There were on average, 142 calls to the flow calculation subroutine per simulation sample in case study 1, with each call lasting 0.0221 seconds. Case study 2 had an average of 20 calls per simulation sample, each lasting 0.0430 seconds. Incorporating variance reduction techniques, to reduce the number of simulations samples, should improve the simulation speed. Depending on the system, additional gains can also be derived from the complete omission of the reconfiguration subroutine. Reconfiguration (the shut-down & restart of nodes) is unnecessary for non-repairable systems and systems for which components are assumed to be statistically independent.

In summary, the size and degree of system activity determine the simulation time. However, as evidenced in Table 3.3, the 21-node non-repairable system with less activity, required less simulation time than the 5-node repairable system, even though 3 times more simulation samples were used for the former. Therefore, the degree of system activity takes precedence over system size, in determining the simulation time.

3.6 Chapter Summary

Complex systems occur in various engineering applications but their reliability analysis is impaired by their very complexity and imposed operational loops. Even in the presence of these impediments, a credible estimate of the relevant reliability and performance indices is required. Diverse assumptions are often invoked to achieve this feat, and though handy, they are also a threat to the correctness of the outcome.

In this chapter, a novel and generally applicable simulation approach to the reliability and performance analysis of complex multi-state systems has been proposed. Unlike existing techniques, it enhances the easy incorporation of system dynamics like flow losses, loading restrictions, and maintenance delays, in the reliability evaluation process. Applicable to repairable and non-repairable systems of any topology, the proposed approach can analyse systems with multiple inputs and outputs, without prior knowledge of their cut sets, path sets, or structure function. The case studies presented have also illustrated it mitigates the need for unrealistic system modelling assumptions.

Like all simulation-based techniques, the approach, however, is computationally intensive. Some practical steps to addressing this problem have, therefore, been suggested in the chapter and will be explored in a future research. The approach, also currently, is limited to static systems but given its flexibility, it can be extended to standby redundant systems, as well as to investigate cascading and common-cause failure models.

Chapter 4

Availability Assessment of Interdependent Multi-state Systems

4.1 Introduction

Realistic engineering systems often possess attributes that complicate their availability assessment. Notable examples being complex topology, multi-state behaviour, component interdependencies, and interactions with external phenomena. For such systems, analytical techniques have limited applicability, and efficient simulation techniques are, therefore, required. A load-flow simulation approach for the reliability analysis of multi-state systems was proposed in Chapter 3. In the approach, each system node is modelled as a semi-Markov stochastic process and the system structure, as a directed graph. An event-driven Monte Carlo simulation is used to reconstruct the random failure and repair events of the system components. As the components go through their cycle of failures and subsequent repairs, their capacities change, and the interior-point algorithm [68] is invoked to determine the performance of the system. The approach employs an adjacency matrix to define the structure of the system and derives the equations of flow across the entire system in the form of matrices. This particularly makes it suitable and intuitive for any system architecture and easily programmable on a digital computer. In terms of applicability, it outperforms other multi-state system reliability analysis approaches, since it does not require state enumeration or cut set definition. It considers realistic system aspects like flow losses, reconfiguration, forced transitions, and multiple competing demands. Load-flow simulation, however, is only applicable to homogeneous independent systems and does not consider restrictions on the number of simultaneous maintenance actions that can take place in the system or limited maintenance teams.

In this chapter, the load-flow simulation proposed in Chapter 3 is extended to support component interdependencies and simplify the availability assessment of realistic engineering systems. The resulting approach is simple and generally applicable to systems, including those with limited maintenance teams, reconfiguration requirements,

and multiple commodity flows. A novel metric for assessing maintenance inadequacy and a real-time component ranking procedure are also introduced. In real-time ranking, failed components are assigned maintenance priorities during simulation in accordance with how much their availability improves system performance and how many idle maintenance teams there are. This eliminates the need for component importance ranking algorithms prior to simulation, which for some systems may be unnecessary. The applicability of the approach is demonstrated by analysing an offshore plant producing oil, gas, and water. The solution obtained is compared against another Monte Carlo simulation-based solution that requires the enumeration of the plant's cut-sets. The proposed approach is shown to be more intuitive, robust to human-induced errors, and require less human effort. Details of the plant and relevant results are published in [56].

The remainder of this chapter is constituted by 6 sections. The next section provides a general overview of the proposed approach, its scope, and novelty. Section 4.3 is dedicated to providing an overview of the relevant modifications to the load-flow approach to include interdependencies. In this section, a generalised procedure for assessing the availability of interdependent multi-state systems is also presented. Details of the simulation procedure and availability assessment algorithms are respectively provided in Sections 4.4 and 4.5. Section 4.6 addresses the availability assessment problem of an offshore multi-commodity plant, which is used to illustrate the systematic roll out of the solution strategy developed in Section 4.3 to a problem of industrial relevance. The usefulness of the new metric for maintenance inadequacy and real-time component ranking are also illustrated here. The implications of the results, efficiency of the approach, and its limitations, climax this section. Finally, the closing remarks; drawing conclusions on the proposed approach constitute Section 4.7.

4.2 Overview of Proposed Approach

Though largely built on the principles proposed in Chapter 3, this work makes a series of new contributions as highlighted thus.

1. A straight-forward procedure for uncoupling interdependencies in systems and an intuitive mathematical model for their adequate representation are proposed. In this chapter, interdependencies refer to cascading events and functional dependencies requiring explicit modelling. CCF will be treated in Chapter 7.

2. Two recursive algorithms are proposed to accurately account for these interdependencies and compute the performance of the system during simulation.

3. To enhance the efficient extraction of system availability and performance indices from the simulation result, easily implementable algorithms have been proposed. Availability, as used here, refers to the ability of a system to function as expected.

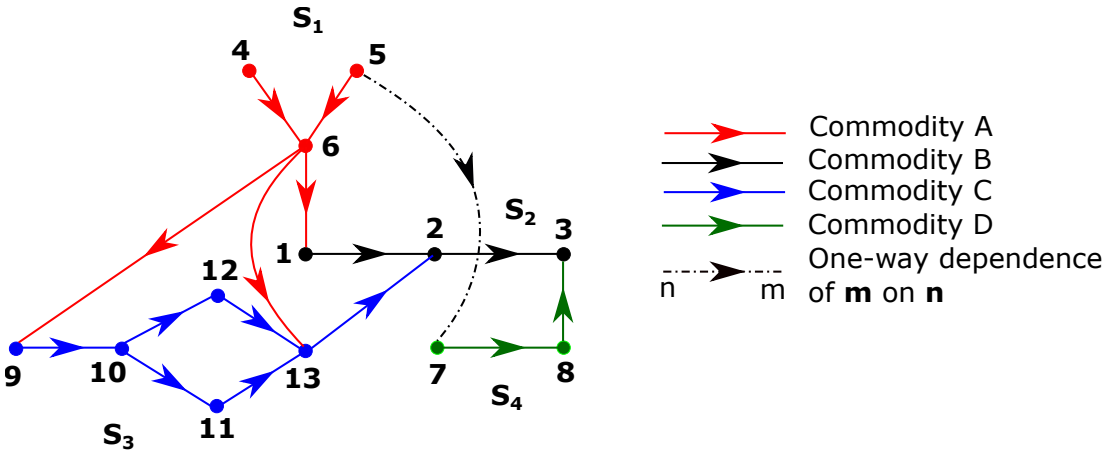


Figure 4.1: An example of a typical interdependent system.

It, therefore, encompasses the reliability, the output characteristics, and the recovery probability of the system after its deviation from expected performance.

4. Reliability and output characteristics are already very common system availability indices. Recovery probability and a new metric for assessing the adequacy of the maintenance process are used as additional system performance indices.

5. A real-time component ranking procedure to identify the sequence of maintenance response that maximises system performance is proposed. The system operator would use this procedure for scenarios dictating preferential maintenance.

6. Finally, a simple but important modification is also made to the original system flow calculation procedure, resulting in appreciable gains in computation time.

In summary, this work extends the applicability of the load-flow simulation approach and improves its computational efficiency.

4.3 Implementation

In this section, the relevant principles governing the modelling of an interdependent system and its components are described. Since these principles are more or less an adaptation of those proposed in Chapter 3, premium is placed only on the necessary modifications. For this purpose, consider the arbitrary system shown in Figure 4.1, which could be a binary-state system or a multi-state flow network [149]. It consists of 4 subsystems and 13 nodes, transporting 4 commodities. The number of subsystems is normally defined by the number of commodities or more generally by the number of closed-loops. This implies, a system could compose of multiple subsystems even when only one commodity type is involved. Nodes 1, 2, and 3, transporting commodity-B,

respectively require commodity, A, C, and D to operate and nodes 9 and 13 in subsystem S_3 rely on flow from subsystem S_1. Also, a certain failure mode of node 5 in subsystem S_1, triggers the partial failure of node 7 in subsystem S_4. This type of interdependency is called a one-way dependence, since the failure of node 5 affects node 7, but state change events in the latter have no effects on the former. The system under review is a perfect example of systems with functional dependencies (the one between nodes 6 and 9, for instance) requiring explicit modelling and cascading dependencies (the one between nodes 5 and 7, for instance). Even for a system this simple, deriving all its cut-sets is time-consuming and error-prone. In the remainder of this section, a generally applicable procedure to overcome these complications is presented.

4.3.1 Decoupling the System

To start the modelling process, all the elements affecting the operation of the system are identified and numbered as illustrated in Figure 4.1. This is followed by the identification and definition of all the node dependencies. Constituent systems (hereafter referred to as subsystems), determined by the different commodities flowing in the system or the number of closed-loops, are assigned subsystem IDs. Each subsystem's associated nodes are identified, graph model developed, and relevant flow equation parameters obtained (see Chapter 3 for details). In identifying the nodes of a subsystem, only those with actual commodity flow are considered. Nodes representing external common-cause initiators, environmental events, or human-system interactions, do not belong to a subsystem. The possible states of each node are then identified and modelled as outlined in Chapter 3.

Consider the system presented in Figure 4.1, with focus on load dependencies. Node 2, for instance, uses commodity-C to drive its operation but transmits commodity-B. One would say it exhibits a dual operation mode, operating both as a sink and a transmission node. The sink mode directly influences flow in S_3, while the transmission mode has a direct influence on flow in S_2. It is, therefore, logical to separate the node into its constituent nodes, each representing a mode of operation. The node representing the sink mode is assigned a new ID while the other retains the ID of the original node. A load-source dependency exists between the nodes, since the transmission node is incapacitated if flow into the sink node is inadequate. They, therefore, make a load-source pair, with the transmission node being the load, and the sink node, the local source. Local sources, otherwise known as support nodes in load-source pairs, are modelled as binary-state objects. State 1, designated active, and assigned a capacity, l, signifies the availability of the dependent node. State 2, with 0 capacity, and designated inactive, depicts otherwise. l is the minimum level of support required to operate the node in the transmission mode, and in practical cases represents the load rating of that component. A $4kW$ rated 3-phase centrifugal pump, for instance, would have its

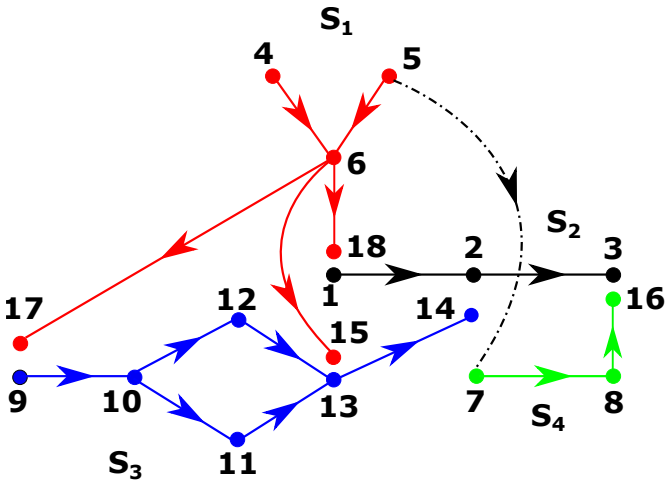


Figure 4.2: Interdependent system showing load-source pairs.

$l = 4kW$. By applying the decoupling procedure described to all load dependency relationships in the system, the following load-source pairs, $\{2, 14\}$, $\{3, 16\}$, $\{1, 18\}$, $\{13, 15\}$, and $\{9, 17\}$, are obtained, as highlighted in Figure 4.2.

To incorporate induced and functional interdependencies in the component model proposed in Chapter 3, two additional parameters, $\mathbb{L}$ and $\boldsymbol{D}$, are introduced. Let i be the index of a node, with $\mathbb{L}_i = \{j, l\}$ defining its load dependency with node j. The dependency defined by $\mathbb{L}_i$ is interpreted as, node i requiring a minimum of l level of flow from node j to operate. When i and j belong to different subsystems, the subsystems are said to be interdependent, since a state change in either node affects flow in both subsystems. If i has load dependency relationships with multiple nodes, $\mathbb{L}_i$ takes the form of a 2-column matrix, with each row defining the node's relationship with another node. Parameter $\boldsymbol{D}_i = \{d_{j1}, d_{j2}, d_{j3}, d_{j4}\}_{u\times 4} \mid j = 1, 2, ..., u-1, u$ defines the single-way causal-effect relationship between node i and other nodes. This type of coupling specifies induced state changes in other nodes following a state change in i. d_{j1} is the state of i triggering the event, d_{j2}; the affected node, d_{j3}; the state the node has to be in to be affected, and d_{j4}; its target state on occurrence of the event. Each row of $\boldsymbol{D}_i$, therefore, defines the behaviour of an affected node, and u, the number of relationships. If i and the affected node, d_{j2}, belong to different subsystems, the subsystem the latter belongs to is dependent on the subsystem of the former. In Figure 4.2, for instance, S_4 depends on S_1, consequent of the relationship between nodes 5 and 7.

Suppose state 3 of node 5 triggers the partial failure (state 2) of node 7. If this happens only when node 7 is in state 1, their dependency is defined by $\boldsymbol{D}_5$ as,

$$\boldsymbol{D}_5 = \begin{pmatrix} 3 & 7 & 1 & 2 \end{pmatrix} \tag{4.1}$$

Equation 4.1 effectively defines the state change induced in node 7 by a state change in node 5. Using the notation described in the preceding paragraph, the expression states

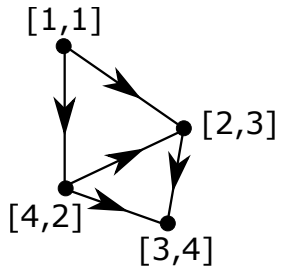


Figure 4.3: Dependency tree for a 4-subsystem system.

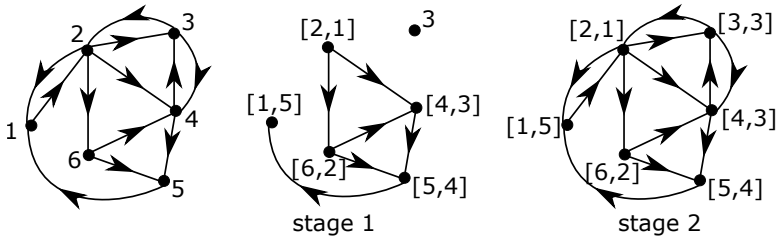


Figure 4.4: Dependency tree: Subsystem ranking procedure.

if node 5 makes a transition to state 3 whilst node 7 is in state 1, the latter is forced through a transition to state 2. Similarly, if a state change in node 7 triggered a state change in node 5 or any other node, $\boldsymbol{D}_7$ would be required to express this.

The final step entails the derivation of the dependency tree relating the subsystems and the ranking of these subsystems according to their position on the tree. Figure 4.3 shows the dependency tree for an arbitrary 4-subsystem system (different from the system in Figure 4.1), where the designation $[a, b]$ specifies that the rank of subsystem a is b. In the ranking procedure, the independent subsystem is chosen as reference and assigned rank 1. The other subsystems are ranked in increasing order of their longest distance from this reference. If two subsystems are interdependent, their mutual link on the tree is discarded, and the ranking done as earlier described. However, if after discarding the mutual links, a node is totally cut off from the rest of the tree, it is assigned the same rank as its dependent pair on the tree. If it is in relationship with multiple nodes, its rank, b, is given by $max\,(\boldsymbol{R})$, $\boldsymbol{R}$ being the set of ranks of all the subsystems it is associated with. Figure 4.4 is an illustration of the procedure, for a system of six subsystems, with interdependencies. Starting with the tree on the left, all mutual links are discarded, leaving node 2 as the only node without a parent. Hence, it is taken to be the reference, and the nodes ranked to complete stage 1 of the procedure. Discarding the mutual links, however, leaves node 3, which is in relationship with nodes 2 and 4, ranked 1 and 3 respectively completely isolated. Node 3, therefore, is assigned rank 3, the maximum of the ranks of the nodes it is connected to.

Let $\boldsymbol{S}_1$, $\boldsymbol{S}_2$, $\boldsymbol{S}_3$, and $\boldsymbol{S}_4$ be the sets of nodes respectively belonging to subsystems $\boldsymbol{S}_1$-$\boldsymbol{S}_4$ of the system presented in Figure 4.1. From Figure 4.2, $\boldsymbol{S}_1 = \{4, 5, 6, 15, 17, 18\}$, $\boldsymbol{S}_2 = \{1, 2, 3\}$, $\boldsymbol{S}_3 = \{9, 10, ..., 14\}$, and $\boldsymbol{S}_4 = \{7, 8, 16\}$. With numbers 1-4 chronolog-

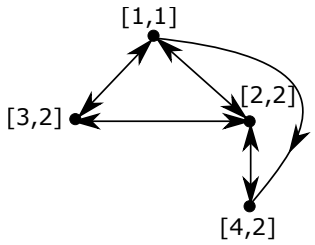


Figure 4.5: Dependency tree for sample interdependent system.

ically assigned to subsystems $\boldsymbol{S}_1$-$\boldsymbol{S}_4$, the system's dependency tree is shown in Figure 4.5. With all the subsystems ranked, an indicator register, $\boldsymbol{I}$, of zeros, such that each element corresponds to a subsystem, is defined. This register indicates (by logic 1 in the relevant position) the subsystem(s) affected by the last node transition.

4.3.2 Accounting for Dependencies

Let $\boldsymbol{\mu}$ be the vector holding the current performance levels of system nodes. When node i makes a transition that results in a change in its performance level, the current capacity of its load-source pair, j, is modified. If $c_{x^-}^{\{i\}}$ is the node's capacity before transition and $c_x^{\{i\}}$, its current capacity, the capacity of node j changes according to Equation 4.2. Where $(\boldsymbol{\mu}, j)$ denotes the j^{th} element of $\boldsymbol{\mu}$.

$$(\boldsymbol{\mu}, j) = c_x^{\{j\}} = \begin{cases} 0 & \text{If } c_{x^-}^{\{i\}} > 0 \text{ and } c_x^{\{i\}} = 0 \\ l & \text{If } c_{x^-}^{\{i\}} = 0 \text{ and } c_x^{\{i\}} > 0 \end{cases} \tag{4.2}$$

A recursive algorithm is required to account for the causal-effect relationships between nodes because of the possibility of nested dependencies. If $\boldsymbol{D}_i$ and x_i are respectively the dependency matrix and current state of node i, the following steps summarise the algorithm;

1. Define a register to hold affected nodes and their target states.
2. Find all nodes affected by the state change (using $\boldsymbol{D}_i$ and x_i) and update the register defined in step 1.
3. Select the last entry, node j, of the register, set its current state to its target state and delete its records from the register.
4. Using $\boldsymbol{D}_j$ and x_j obtained in step 3, in place of $\boldsymbol{D}_i$ and x_i, repeat steps 2 and 3.
5. Repeat steps 2 through 4 until the register defined in step 1 is empty.

On each node transition, $\boldsymbol{\mu}$ is updated and any load dependencies accounted for, as described by Equation 4.2.

4.3.3 Node Reconfiguration

Node i is shut down if its flow falls to or below its threshold, Λ_i, or if flow through its load-source pair, j, falls below l. If a node is shut down, its current and next states are saved, its next transition time set to ∞, and its current capacity, to 0. When the condition leading to shut-down is resolved, the node is restarted and restored to its previous state. The period, t_{spent}, spent in shut-down is accounted for by shifting its next transition time to $t_{next} + t_{spent}$, where t_{next} is its transition time before shut-down. This time shifting is repeated also for the node's next preventive maintenance due time, if its preventive maintenance interval is a function of the time spent in operation.

In practice, maintenance durations are not affected by node shut-down events. Therefore, if the next state of a node is superior in performance and reliability to its current, only its current capacity is modified. Modifying its next transition time would mean delaying its restoration, which may negatively affect the simulation outcome. Algorithms for shut-down and restart of nodes are presented in Chapter 3, which, however, should be modified to incorporate component interdependencies.

Let $\boldsymbol{\delta}$ be the set of nodes currently in shut-down state and $\boldsymbol{\eta}$, the vector of system node flows. Node i is added to $\boldsymbol{\delta}$ if and only if its shut-down is due to the condition, $(i, \boldsymbol{\eta}) \leqslant \Lambda_i$. As a rule-of-thumb, nodes that do not satisfy the threshold flow condition are shut down first. Next, flows through sink nodes that have load-source pairs are assessed. If for a sink node, j, $(\boldsymbol{\eta}, j) < l$, its load-dependent node, i, is shut down. The same order is followed for node restart, where node flows are assessed for satisfaction of the relevant conditions. If the condition $(\boldsymbol{\eta}, j) = l$ is met for a sink node, its load-dependent pair, i, in shut-down is restarted.

4.3.4 Determining system performance at time t

The goal of system analysis is to determine the amount of commodity flow through output nodes. This, in turn, requires that flow is calculated after every transition that results in a performance level change of a node. Given node interdependencies, a state change in one node may give rise to state changes in a series of other nodes. The system may go through a number of performance levels in the process, but the effective performance is the one attained after the last transition. A recursive algorithm, therefore, is employed for this purpose during system simulation, as outlined thus;

1. Define $\boldsymbol{\mu}_t$; a temporary variable and set its value to $\boldsymbol{\mu}$ (i.e., $\boldsymbol{\mu}_t = \boldsymbol{\mu}$). Where $\boldsymbol{\mu}$ is the vector of current node capacities.

2. In $\boldsymbol{\mu}_t$, set the capacities of all the nodes in $\boldsymbol{\delta}$ to their values before shut-down. This step is required to determine which nodes in shut-down can be restarted.

3. Using $\boldsymbol{I}$, select the highest ranked subsystem which indicator is 1 and calculate its flow using $\boldsymbol{\mu}_t$. The highest ranked subsystem corresponds to the subsystem

with the smallest rank. If multiple subsystems meet this requirement, randomly select a candidate. Go to step 8 if there are no non-zero elements in $\boldsymbol{I}$.

4. Set in $\boldsymbol{I}$, the indicator for the subsystem in step 3 to 0.

5. Restart and shut down nodes according to the procedure outlined in Section 4.3.3.

6. Following a node transition, the subsystem hosting the node is identified, and its position in $\boldsymbol{I}$ set to 1. This is only required if the shut-down or restart of the node is a direct effect of a state change in its load-source pair (see Section 4.3.1).

7. Repeat steps 1 to 6, making sure interdependencies are accounted for and $\boldsymbol{\mu}$ updated on every transition.

8. Get the flows through the output nodes, save as a function of time, and terminate algorithm.

4.4 The System Simulation Procedure

Simulation normally entails repeated calculation of system output, as nodes undergo their transition cycles. Calling the interior-point algorithm for every transition, as proposed in [55], may impose unprecedented computational burden. This is because, a certain system configuration may be attained more than once, making multiple calculations for the same configuration a possibility. To overcome this problem, it is desirable to determine node flows for all the possible combinations of system node performance levels prior to simulation. Let $\boldsymbol{\beta}$ be the matrix holding these combinations and $\mathbf{C}_u^{\{i\}}$, the set of unique performance levels of node i. $\boldsymbol{\beta}$ is an $M \times \prod_i^M n^{\{i\}}$ matrix, M being the total number of nodes, excluding external nodes, and $n^{\{i\}}$, the number of unique performance levels of node i. For instance, if the capacity of node 1 is defined by $\mathbf{C} = \{10, 20, 0, 0, 10\}$, $\mathbf{C}_u^{\{1\}} = \{0, 10, 20\}$ and $n^{\{1\}} = 3$. For each combination of performance levels, the corresponding node flows are calculated and recorded in a second matrix, $\mathbf{F}$. During simulation, $\boldsymbol{\beta}$ is searched for the combination of node performances corresponding to the current system configuration, and its pre-stored node flows in $\mathbf{F}$ simply read off. By this, flow calculation for every configuration is carried out only once. It, however, is worthwhile noting that for large systems, $\boldsymbol{\beta}$ gets prohibitively large, such that the time to search for a configuration exceeds its flow calculation time. The search time, however, can be reduced by smart allocation and search procedures. Therefore, it is advised that the average times to complete both procedures are compared prior to simulation.

For an interdependent system like the one in Figure 4.1, the procedure described above is carried out for each subsystem, and in each case, only nodes belonging to that subsystem are considered.

4.4.1 Forcing Maintenance

Algorithm 3 Forcing maintenance: Limited dedicated maintenance teams.

Require: n_1, λ_1, n_2, λ_2, $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$

```
1:  k ← 1                                              ▷ initialize indicator
2:  while k ⩽ 2 do
3:      v ← n_k − λ_k                                  ▷ get idle teams
4:      while v > 0 and h_k ≠ ∅ do
5:          select node according to priority
6:          make maintenance state the current state x
7:          sample next transition using x
8:          delete node from h_k
9:          v ← v − 1,   λ_k = λ_k + 1
10:     end while
11:     k ← k + 1
12: end while
```

With limited maintenance teams, the commencement of maintenance is not always instantaneous. Therefore, the transition from a degraded state or to Preventive Maintenance (PM) has to be manually executed during simulations. Let n_1 denote the number of teams dedicated to Corrective Maintenance (CM), n_2, the number of Preventive Maintenance teams, λ_1, the number of busy CM teams, and λ_2, the number of busy PM teams. Following its transition, a node is added to the set, $\boldsymbol{h}_1$, of nodes requiring repairs if its new state is directly linked to a CM state, and to $\boldsymbol{h}_2$, if its PM is due. At time, t, maintenance is forced if there are idle maintenance teams and $\boldsymbol{h}_1$ or $\boldsymbol{h}_2$ is not empty. This procedure is described by Algorithm 3, where $k = 1$ and $k = 2$ respectively denote CM and PM.

If PM is carried out only when the system is perfect, the algorithm is terminated after the task for $k = 1$ if at least one of $\boldsymbol{h}_1 \neq \varnothing$, $\lambda_1 > 0$, and $\lambda_2 > 0$ is true. Each of these conditions means either there is a failed component waiting to be repaired or maintenance is in progress, any of which suggests the system is not in a perfect state. In most applications, the PM of a node is delayed if it is in a degraded state, until after CM. If this is the case, a node belonging to both $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$ is rejected when selected during the PM task ($k = 2$) of the algorithm. Also, most systems are assumed to operate under a perfect maintenance scenario. This means, nodes are repaired to an as-good-as-new condition, making any pending PM tasks for a repaired node no longer necessary. In such cases, a node's records are deleted from both $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$ after CM.

Algorithm 3 is based on the assumption that CM and PM are carried out by different teams (dedicated maintenance). It is, however, adaptable to systems where the same team can carry out both maintenance actions (shared maintenance). Let n_t be the total number of maintenance teams, λ_t, the number of busy maintenance teams, and

$\boldsymbol{h}_t = (\boldsymbol{h}_1 \cup \boldsymbol{h}_2)$, the set of nodes in the maintenance queue. Algorithm 4 outlines the procedure for forcing maintenance in shared maintenance scenarios.

Algorithm 4 Forcing maintenance: Limited shared maintenance teams.

Require: n_t, λ_t, $\boldsymbol{h}_t$

1: $v \leftarrow n_t - \lambda_t$ ▷ get idle teams
2: **while** $v > 0$ **and** $\boldsymbol{h}_t \neq \varnothing$ **do**
3: select node according to priority
4: make maintenance state the current state x
5: sample next transition using x
6: delete node from queue ($\boldsymbol{h}_1$ or $\boldsymbol{h}_2$)
7: $v \leftarrow v - 1, \quad \lambda_t = \lambda_t + 1$
8: **end while**

4.4.2 Maintenance Priority & Real-time Component Ranking

In practical applications, system availability is computed based on some predefined maintenance priority ranking of nodes [161]. However, for complex systems, node priority ranking is either impossible, time-consuming, error-prone, or requires component importance ranking algorithms [49,69]. Priority maintenance is required to reduce the *maintenance response inadequacy* for critical nodes, a measure of how long a node waits in the maintenance queue. It depends on the number of CM teams, the failure characteristics of nodes, and the efficiency of the maintenance teams. This implies, a system may have fewer CM teams than nodes and still not require priority maintenance if node failures are rare or maintenance durations are relatively short. For such systems, pre simulation priority ranking is a mere waste of time and computing resources. In view of this, real-time node ranking, where nodes are ranked during simulation is proposed.

Prior to forcing maintenance, failed nodes are arranged into groups, based on the number of available CM teams. Let $v_1 = n_1 - \lambda_1$ be the number of idle CM teams at time t, and g, the number of failed nodes. If there are more failed nodes than there are available CM teams (i.e., $v_1 < g \mid v_1 \neq 0$), all the possible combinations of nodes that can be repaired are generated, producing $\binom{g}{v_1}$ groups, each containing v_1 failed nodes. The nodes in the first group are temporarily set to their expected performance levels after CM, whilst those in the other groups remain in their current states. The expected system performance, given the new node states is determined, and the procedure repeated for all the node groups. The maintenance of the nodes in the group with the best system performance is then initiated. Real-time ranking, therefore, does not only take into consideration the current system conditions but happens only when necessary, the latter being a computational efficiency boost, especially for complex systems. This ranking procedure is applicable to shared maintenance scenarios, as well.

4.4.3 The Simulation Algorithm

With the relationships between system nodes defined, a Monte Carlo simulation algorithm is required to reconstruct the operation of the system and derive its availability indices. An efficient event-driven and non system-specific simulation algorithm is proposed for this purpose. It proceeds by going through sampled node transitions, determining the flows through the output nodes, and collecting these flows. Starting with the nodes in their initial states at time, $t = 0$, the system's initial performance is determined and the next transition times of nodes sampled. The simulation jumps to a new time, $t = t_{min}$, where t_{min} is the minimum of the next transition times of nodes. Nodes with transition times equal to t_{min} are identified, the required state changes effected, their next transition times sampled, the new system performance deduced, and the next simulation jump determined. This continues until the mission time, T_m, is exceeded. The relevant availability and performance indices are derived at the end of the simulation from the saved system performance history. To ease the computational burden, flow is calculated only if at time, t, the current and previous system node capacity vectors ($\boldsymbol{\mu}$ and $\boldsymbol{\mu}_{old}$) are different. The simulation procedure is summarised as follows.

1. Initialise the register to store the output node history and calculate flows across all the subsystems, as described in Section 4.4. Define the mission time, T_m, number of simulation samples, N, and number of maintenance teams, (n_1, n_2).

2. Define $\boldsymbol{\mu}$ from the initial states and capacities of nodes. Set $\boldsymbol{\tau} = \{\infty\}^M$, $\boldsymbol{\mu}_{old} = 0$, $\boldsymbol{\delta} = \boldsymbol{h}_1 = \boldsymbol{h}_2 = \varnothing$, $t = \lambda_1 = \lambda_2 = 0$, and all the elements of $\boldsymbol{I}$ to zero.

3. Sample the next transition time for all the nodes and update $\boldsymbol{\tau}$. Also determine their PM due times (if applicable), and store in $\boldsymbol{\tau}_{pm}$.

4. Identify the nodes with transition time equal to t, updating their current states, $\boldsymbol{\mu}$, $\boldsymbol{h}_1$, and accounting for any interdependencies.

5. For each node in step 4, determine its subsystem and set the indicator element for the subsystem in $\boldsymbol{I}$ to 1. Sample the node's next transition and update $\boldsymbol{\tau}$. If it is just from PM, determine its next PM due time and update $\boldsymbol{\tau}_{pm}$.

6. Determine nodes whose value in $\boldsymbol{\tau}_{pm}$ equals t and add them to $\boldsymbol{h}_2$.

7. Force maintenance as described in Section 4.4.1 if there are nodes in the maintenance queue and provided there are available maintenance teams. That is, $\boldsymbol{h}_1 \neq \varnothing$ and $n_1 - \lambda_1 > 0$ or $\boldsymbol{h}_2 \neq \varnothing$ and $n_2 - \lambda_2 > 0$.

8. If $\boldsymbol{\mu}_{old} \neq \boldsymbol{\mu}$, determine the system performance, as outlined in Section 4.3.4.

9. Set $\boldsymbol{\mu}_{old} = \boldsymbol{\mu}$ and determine the next system transition time, t, given by the minimum of all the possible transition times. The next simulation jump, therefore, occurs at time, $t = min(min(\boldsymbol{\tau}), min(\boldsymbol{\tau}_{pm}))$.

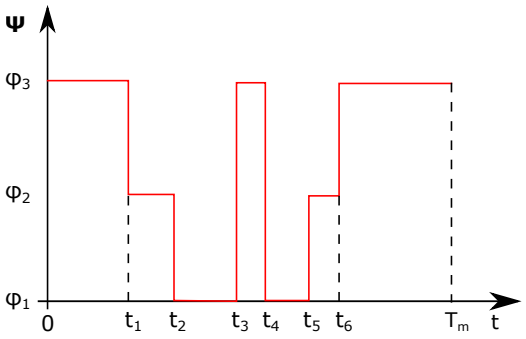


Figure 4.6: System performance history for one Monte Carlo realisation.

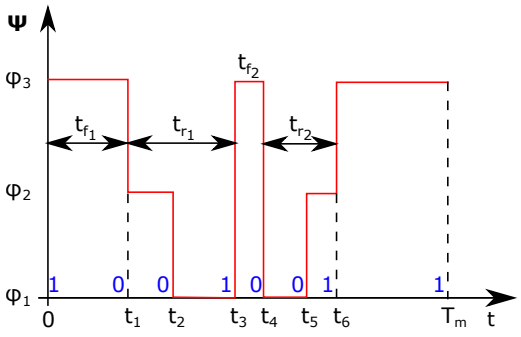


Figure 4.7: System performance history showing failure and recovery times.

10. Repeat steps 4 through 9 until t exceeds T_m.

11. Repeat steps 2 through 10, N times and compute the relevant availability indices.

4.5 Obtaining the Availability and Performance Indices

Details on the frequently used availability and performance indices are available in many standard reliability texts [27,77]. However, the following indices; *reliability*, *recovery probability*, *instantaneous availability*, *steady-state availability*, *instantaneous output*, and *expected output* have been considered, for completeness.

$$\boldsymbol{\Psi} = \{\psi_i\}^j \mid \psi_i \in \boldsymbol{C}, \quad \mathbb{T} = \{t_i\}^j \mid 0 < t_i \leqslant T_m \tag{4.3}$$

During system simulation, a node transition results in system transition only if it leads to the attainment of a new system performance level. The main task, therefore, is the collection of these performance levels and their corresponding attainment times. For a simulation sample, let the system performances be stored in $\boldsymbol{\Psi}$ as they are attained and the corresponding transition times, in $\mathbb{T}$. If these are defined according to Equation 4.3, where t_i is the i^{th} transition time, ψ_i, the corresponding system performance, and j, the total number of system transitions. $\boldsymbol{C} = \{\phi_1, \phi_2, ..., \phi_n\}$ is the set of possible system performances obtained from the simulation, where $\phi_z \mid z \in \{1, 2, ..., n\}$ is the z^{th} system performance level and n, the number of possible performance levels.

Figure 4.6 shows the performance history of a hypothetical 3-performance-level system. A system simulation of N samples contains N such histories, which are used to derive the various reliability and availability indices of the system. The algorithms proposed for this purpose are based on an efficient binary-state translation of the system simulation history. Translating the multi-state performance history, $\boldsymbol{\Psi}$, to a binary string effectively reduces the multi-state reliability problem to its simpler binary-state counterpart. This enhances the application of well-known binary-state system reliability algorithms to the calculation of multi-state system reliability and performance indices.

4.5.1 System Reliability and Recovery Probability

The reliability of a repairable system at time, t, is the probability that the system will not fail between times 0 and t, given it was new or repaired to as-good-as-new at time 0. Failure is relative, and depends on the type of system and the success criteria set by the analyst. For multi-state systems, it is defined with respect to the system operating below a certain performance level. The likelihood that the system will be restored to this performance level in time, t, after failure, is defined by its recovery probability, $r(t)$.

Consider the system represented by Figure 4.6, and let it be considered failed when operating below ϕ_3. Figure 4.7 shows its performance history for one simulation sample, with annotations portraying the physical meanings of failure and recovery. t_{f_i} is the time the system takes at ϕ_3 before the i^{th} failure, and t_{r_i}, the corresponding recovery duration. If the transitions resulting in system performance of ϕ_3 are replaced with 1, and 0, otherwise, the system performance history is translated into a string of 1's and 0's, as shown in Figure 4.7. This string is used in conjunction with $\mathbb{T}$ to derive the reliability and other performance indices of the system. For instance, system failures and recovery are easily identified by the positions of the sub strings, '$1-0$' and '$0-1$', respectively, in the string. If these positions are defined by vectors $\boldsymbol{\sigma}_f$ and $\boldsymbol{\sigma}_r$, the sets of system failure and corresponding recovery times can be obtained from $\mathbb{T}$. These sets respectively define the failure and recovery time density functions, $f(t_f)$ and $f(t_r)$, of the system. Their corresponding cumulative density functions, $F(t_f)$ and $F(t_r)$, are used to deduce $R(t)$ and $r(t)$ from $1-F(t_f)$ and $1-F(t_r)$, respectively.

For a multi component system assumed initially perfect (which is the case if reliability is of interest), only the first failure times, t_{f_1}, of each simulation sample are used to define $f(t_f)$. This is explained by the fact that, depending on the structure and properties of its components, a system may have one or more components in a degraded state and still attain/maintain the required performance. It is the case, for example, in a 2-branch, purely parallel system, where, one branch is sufficient to attain nominal system performance. Though both yield the same performance, the system with only one branch available has a higher probability of failure. System failure times yielded in this case, therefore, underestimate the reliability of the system. A system's performance history alone cannot say exactly the states of its nodes at system recovery. It is, therefore, impossible to determine whether the higher order failure times, $t_{f_2}, t_{f_3}, ..., t_{f_j}$, were yielded by the perfect system. Though this is possible by collecting the system state vector at every transition, it is a computationally expensive option. Hence, system reliability, redefined as a function of first failures only, is expressed as, $R(t) = 1-F(t_{f_1})$.

A system's reliability and recovery probability have been shown to be derived from the cumulative density functions of its failure and recovery times, respectively. These density functions are directly obtainable from the system performance history via an approximation technique. With the failure and recovery times collected, the mission

time, T_m, can be divided into time-steps, δ, and the average contribution of each failure time, t_{f_i}, and recovery time, t_{r_i}, to each time-step estimated. The accuracy of the estimates is determined by how small δ is, relative T_m. It is, however, worthwhile noting that the discretisation is only required to estimate the instantaneous performance indices, the actual simulation does not require time-steps. Outlined below are the steps for deducing $R(t)$ and $r(t)$ from the system performance history.

1. Define δ_t; the total number of time-steps, such that $\delta_t = \lceil T_m/\delta \rceil$ and ϕ_{ref}, the performance of interest. Set $R(t) = r'(t) = \{0\}^{\delta_t}$, $i = 1$ and $c = 0$, where c is the number of recovery times computed, and i, the index of the current history.

2. Given $\boldsymbol{\Psi}_i$ and $\mathbb{T}_i$ are the performance history of the i^{th} simulation sample, modify their contents to include system performances at $t = 0$ and $t = T_m$. The performance, ψ_j, at the last system transition is made the performance at $t = T_m$.

3. Define a binary string, $\boldsymbol{\varpi} = \{\varpi_k\}^j \mid k = 1, 2, ..., j$, such that $\varpi_k = 1$ if $\psi_k \geqslant \phi_{ref}$, and 0, otherwise.

4. Obtain $\boldsymbol{\sigma}_f$ and $\boldsymbol{\sigma}_r$; the locations of failures and recoveries in $\boldsymbol{\varpi}$ with their corresponding times, $\mathbb{T}_f$ and $\mathbb{T}_r$, such that $\mathbb{T}_f = \left(\mathbb{T}, \boldsymbol{\sigma}_f + 1\right)$ and $\mathbb{T}_r = (\mathbb{T}, \boldsymbol{\sigma}_r + 1)$.

5. Take the first element of $\mathbb{T}_f$, determine j_0; the number of time-steps it represents and increment the first j_0 elements of $R(t)$ by 1. That is, $(R(t), 1 \rightarrow j_0) = (R(t), 1 \rightarrow j_0) + 1$; where, $j_0 = \lceil (\mathbb{T}_f, 1)/\delta \rceil$.

6. Discard the last element of $\mathbb{T}_f$ if it has more elements than $\mathbb{T}_r$, and determine the recovery durations, $\boldsymbol{\Delta}_r$, such that, $\boldsymbol{\Delta}_r = \mathbb{T}_r - \mathbb{T}_f$.

7. Take the first element of $\mathbb{T}_r$, determine j_0 and increment the first j_0 elements of $r'(t)$ by 1, such that, $(r'(t), 1 \rightarrow j_0) = (r'(t), 1 \rightarrow j_0) + 1$ and $j_0 = \lceil (\boldsymbol{\Delta}_r, 1)/\delta \rceil$. Also increment the recovery time counter, c, by 1, such that, $c = c + 1$.

8. Repeat step 7 until all the elements of $\boldsymbol{\Delta}_r$ have been covered, and increment the simulation index counter, i, by 1, such that, $i = i + 1$.

9. Repeat steps 2 to 8 until $i = N + 1$, N being the number of simulation samples.

10. Compute the reliability as, $R(t) = R(t)/N$, the non-recovery probability, as $r'(t) = r'(t)/c$, the recovery probability as $1 - r'(t)$, and terminate the algorithm.

4.5.2 Instantaneous Availability and Expected System Output

The instantaneous availability, $A(t)$, of a system is the probability that its performance at time, t, is greater than or equal to some reference, ϕ_q. The expected system performance at this time defines the instantaneous output, $X(t)$.

Let $\boldsymbol{P}(t) = \{p_z(t)\}^k$ be the vector of instantaneous state probabilities; the probabilities of the system being in each of its performance levels at time, t. If the elements of $\boldsymbol{C}$ are ordered such that, $\phi_z < \phi_{z+1} < ... < \phi_k$, $A(t)$ and $X(t)$ are defined as follows.

$$A(t) = \sum_{z \geqslant q}^{k} p_z(t), \quad X(t) = \sum_{z=1}^{k} \phi_z p_z(t) \tag{4.4}$$

The average values of $A(t)$ and $X(t)$ over the mission time, respectively define the steady-state availability and the expected system output. They are obtained from Equation 4.4 by replacing the instantaneous state probabilities, $p_z(t)$, with their average values, p_z. Where, p_z is the fraction of the mission time spent at performance level z. Therefore, obtaining a system's state probabilities is key to its availability and performance assessment. The following steps describe how this is efficiently achieved.

1. Define δ_t; the number of time-steps, n; the number of performance levels, and set $z = 1$. Where, z is the system performance level under consideration.

2. Set $p_z(t) = \{0\}^{\delta_t}$, $\tau_z = 0$, and $i = 1$. Where, τ_z is the total time spent at performance level z and i, the index of the current simulation history.

3. Modify $\boldsymbol{\Psi}_i$ and $\mathbb{T}_i$ to include system performances at $t = 0$ and $T = T_m$.

4. Define a binary string, $\boldsymbol{\varpi} = \{\varpi_k\}^j \mid k = 1, 2, ..., j$, such that, $\varpi_k = 1$ if $\psi_k = \phi_z$, and 0, otherwise.

5. Set the last element of $\boldsymbol{\varpi}$ to 0. This ensures the period between the last transition and T_m is accounted for, given the transition was to performance level z.

6. Obtain $\boldsymbol{\sigma}$; the locations of the sub string, '$1 - 0$' in $\boldsymbol{\varpi}$. Compute $\mathbb{T}_1 = (\mathbb{T}, \boldsymbol{\sigma})$, $\mathbb{T}_2 = (\mathbb{T}, \boldsymbol{\sigma} + 1)$, and τ_z, such that, $\tau_z = \tau_z + \sum(\mathbb{T}_2 - \mathbb{T}_1)$.

7. Deduce j_1 and j_2; the number of time-steps the first elements of $\mathbb{T}_1$ and $\mathbb{T}_2$ respectively represent and increment elements j_1 to j_2 of $p_z(t)$ by 1.

8. Repeat step 7 for the remaining elements of $\mathbb{T}_1$ and $\mathbb{T}_2$, and set $i = i + 1$.

9. Repeat steps 3 to 8 until $i = N + 1$, compute $p_z = \tau_z / NT_m$, $p_z(t) = p_z(t)/N$, and set $z = z + 1$.

10. Repeat steps 2 to 9 until $z = n + 1$ and terminate the algorithm.

With the system state probabilities known, $A(t)$ is obtained from Equation 4.4. Often, however, only the availability with respect to a set of states or a range of performances is required. In both cases, deriving all the system state probabilities is unnecessary and time-consuming. The good news is that, the algorithm described above remains applicable, albeit with minimal modifications. The availability with respect to a given

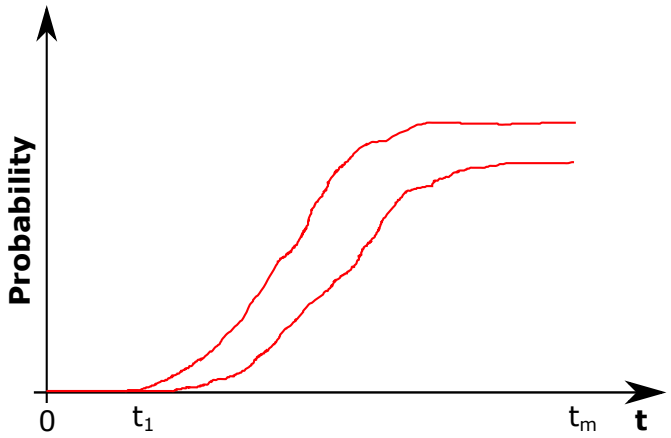


Figure 4.8: Bounds on maintenance response inadequacy of a sample system.

condition is obtained by disregarding step 10 and modifying step 4 to reflect the desired condition (s). For instance, the availability with respect to system performance being between 10 and 20 would be implemented as, $\varpi_k = 1$ if $10 \leqslant \psi_k \leqslant 20$ in step 4.

4.5.3 Maintenance Response Inadequacy

When there are as many maintenance teams as repairable nodes, the latter spend negligible time in failed or degraded states before maintenance intervention. This is not the case with limited maintenance teams, as failed nodes would have to wait in the maintenance queue until a maintenance team is available. The probability that at time, t, a node is in the maintenance queue defines the system's maintenance response inadequacy relative to that node. It is a measure of how severe the effect of limited maintenance is on the node's availability and contribution to system performance.

With the basis for increasing the maintenance team size established, a system's maintenance response inadequacies can be used to determine which nodes to prioritize, assuming certain node repairs require specific specialist skills. The maintenance response inadequacy of a node is a right-continuous increasing function that approaches a steady-state value with time. It is obtained by adding the instantaneous state probabilities of the node's repairable failed and degraded states. At the system level, the maintenance response inadequacies of a system's nodes can be combined, and a bound on the probability of at least one of them being in the maintenance queue obtained. These bounds provide a means of comparing the efficiency of two maintenance teams without explicit reference to system performance. They also indicate when, during the mission a maintenance team size scale-up is actually necessary, an attribute useful to ageing systems. Figure 4.8, for instance, suggests maintenance team size scale-up for the sample system is necessary only after $t = t_1$.

Though the maintenance response inadequacy indicates the need to increase the maintenance team size, it does not state its actual effect on system performance. In order

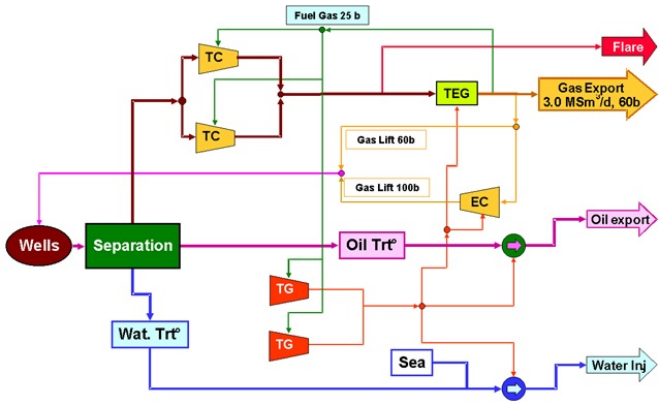
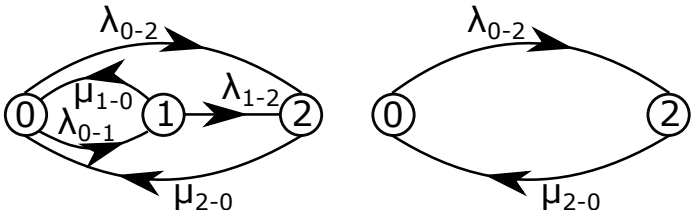


Figure 4.9: Schematic of an offshore installation.



(a) TCs and TGs (b) TEG and EC

Figure 4.10: State-space diagrams of components.

to justify this need under cost constraints, its effect on system performance should be established. This entails comparing the performances yielded by the system under zero maintenance response inadequacy and with the current maintenance team size. Zero maintenance response inadequacy is obtained by setting the number of maintenance teams to infinity. The difference in performance represents the maximum achievable gain from scaling up the maintenance team. Its monetary value can be obtained and compared against the minimum maintenance team scale-up cost, thereby enhancing a robust decision-making process.

4.6 Case Study: An Offshore Oil Installation

Using the system originally presented in [161], the applicability of the proposed approach to interdependent systems prone to limited maintenance teams is illustrated here.

4.6.1 Problem Formulation

Figure 4.9 shows the schematic of an offshore installation which failure and repair transitions of six of its components are described by Figure 4.10. The remaining components are assumed to be perfectly reliable and the notations in Figures 4.9 and 4.10 are defined thus, TG; Turbo Generator, TC; Turbo Compressor, TEG; Try-ethylene Glycol

Table 4.1: Component repair priority.

Priority	Component	System Condition
1	TEG	-
	TG	other TG unavailable
	TC	other TC unavailable
2	EC	-
	TC	other TC available
3	TG	other TG available

Unit, EC; Electro-Compressor, λ_{m-n}; failure rate from state m to n, and μ_{m-n}; repair rate from state m to n. State 0, in Fig. 4.10 denotes the relevant component in its normal operating mode, and state 1, its partial failure mode. When partially failed, the component maintains its nominal performance but with an increased failure probability to state 2, where it is completely failed.

The Well nominally produces 5.0×10^6 units of gas, 26500 units of oil, and 8000 units of water a day. These are separated at the Separation Unit and transmitted via independent dedicated paths, as shown in Figure 4.9. The nominal gas demand is 3×10^6 units at 60bar, and when gas production exceeds demand, for safety reasons, the excess is burnt as flare. Additional details on the offshore plant are available in [161].

4.6.1.1 Interdependencies & Reconfiguration

The major components of the plant (TEG, EC, oil pump and water pump) require continuous supply of electricity to function. This reliance creates a functional coupling between the electricity network and the paths transporting the three commodities. A second functional coupling is introduced by the reliance of TCs and TGs on dried compressed gas, for their functioning. Each TG is rated 13MW, the TEG and EC consume 6MW each, while the two pumps consume 7MW each. When only one TG is available, the EC and the water pump are shut down to maintain the production of oil and gas. To ensure nominal production, a fraction of dried compressed gas (1.0×10^6 units) is diverted, compressed by the EC to 100bar, and re-injected into the Well. If the EC is unavailable, the gas is injected directly into the well at 60bar, resulting in a production at 80% of nominal levels. With no gas injection, production drops to 60%.

4.6.1.2 Maintenance Policy

Corrective maintenance is carried out according to a predefined priority, as expressed in Table 4.1. Repairs, once initiated, remain unaffected by the failure of other components, regardless of their superiority on the priority list. In addition to corrective maintenance, TCs and TGs undergo three types of preventive maintenance interventions, while the EC undergoes one. To ensure minimal effect on performance, preventive maintenance is carried out only when the system is perfect. Table 4.2 outlines the various PM

Table 4.2: Component preventive maintenance schedule.

PM Type	Component	Interval(h)	Mean Duration(h)
1	TC,TG	2160	4
2	EC	2666	113
3	TC,TG	8760	120
4	TC,TG	43800	672

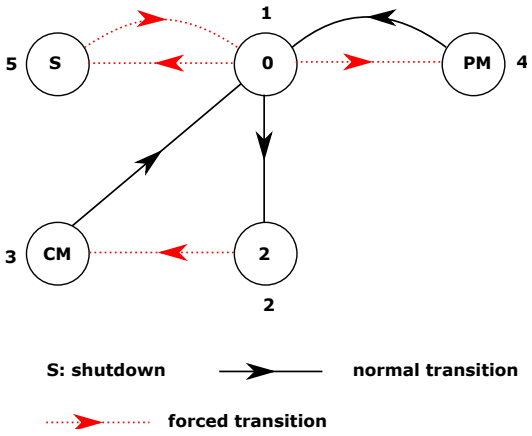


Figure 4.11: State-space for EC and TEG.

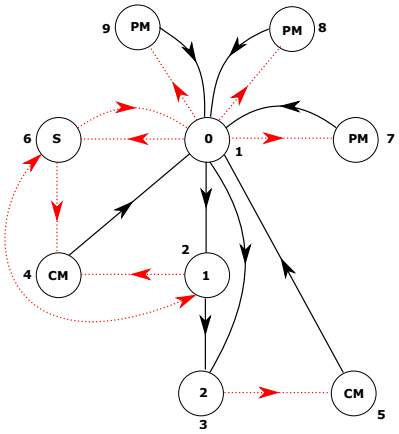


Figure 4.12: State-space for TC and TG.

types, their intervals, and mean duration. The latter are assumed to be exponentially distributed, and the former, as the absolute time between successive PM interventions.

4.6.1.3 Monte Carlo Simulation

In [161], the goal was to determine the production availability of the plant under the maintenance policy described. It was approached by enumerating the plant's production levels, reconstructing the cycle of component failures & maintenance, and monitoring production level occurrences. Identifying the production level corresponding to a given plant configuration during the simulation had required the use of an innovative approach based on cut sets. In practice, each production level is identified by a pair of cut sets defined as minimum and maximum cut sets. Although the solution proposed was very efficient and innovative, it required the manual identification of those cut sets and their corresponding production levels. This, even for a moderately sized system, can be quite time-consuming, error-prone, and may become impractical for some systems.

4.6.2 Solution Procedure

Given the challenges of the Monte Carlo simulation approach used by the original authors, the approach proposed in this chapter was applied to the plant.

Figures 4.11 and 4.12 are modifications of the state diagrams of the plant's components presented in Figure 4.10. The modifications are such that the realistic operation

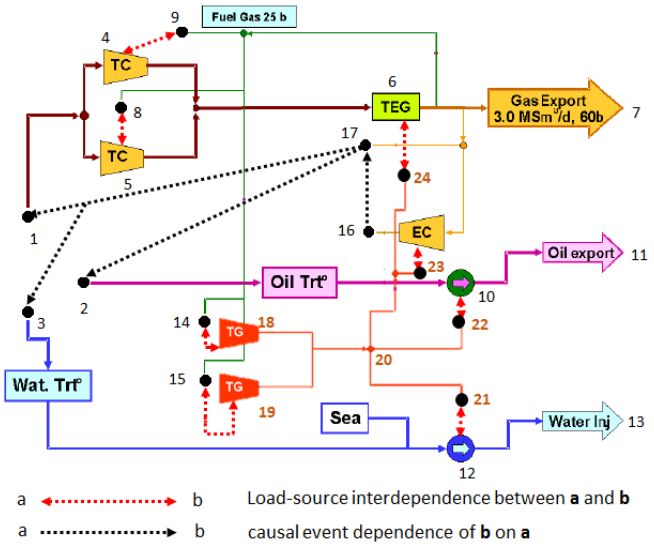


Figure 4.13: System model showing dependencies.

of components, consequent of system dynamics is reflected. The state, shut-down (S), is introduced to account for restart and shut-down (reconfiguration of the component). In the plant, PM takes place only when all its components are in their perfect states. This explains why the transition to PM in the modified state diagrams is from state 0. With the exception of state 4 in Figure 4.12, a component has 0 capacity when in any of the shut-down, CM and PM states. State 4 is an exception because it represents a minimal repair, and, therefore, does not require the component to be taken out of operation. Hence, its capacity from state 2 is retained. Transitions to the maintenance states, CM and PM, are forced, as they depend on the availability of an idle maintenance team. A component for instance, remains in state 3 indefinitely until an idle maintenance team initiates its repair. Similarly, transitions to shut-down are forced, since they denote an induced unavailability of a component due to the unavailability of another component.

$$\mathbf{D}_{16} = \begin{pmatrix} 1 & 17 & 1 & 2 \\ 2 & 17 & 2 & 1 \\ 5 & 17 & 2 & 1 \end{pmatrix}, \quad \mathbf{D}_{17} = \begin{pmatrix} 1 & 1 & 1 & 2 \\ 1 & 2 & 1 & 2 \\ 1 & 3 & 1 & 2 \\ 2 & 1 & 2 & 1 \\ 2 & 2 & 2 & 1 \\ 2 & 3 & 2 & 1 \end{pmatrix} \tag{4.5}$$

4.6.2.1 Modelling the Plant

Shown in Figure 4.13 is the plant's schematic, with the relevant nodes and their relationships. The Well is separated into three nodes, 1, 2, and 3, each supplying gas,

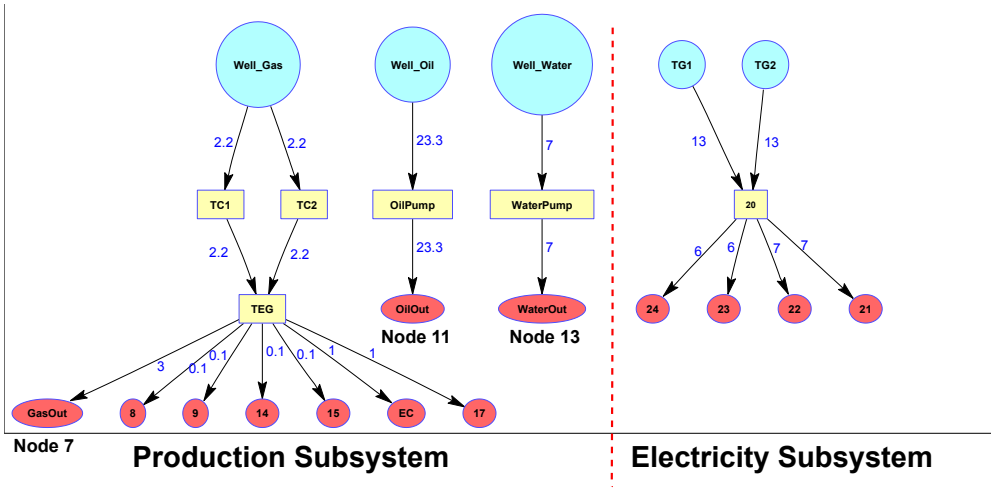


Figure 4.14: Plant network model.

oil, and water respectively. Its third production level is unreachable, since there is no gas lift only if the entire system is failed. Therefore, each of the three nodes exists in only two states, 100% (state 1) and 80% (state 2) nominal output. A third state, with capacity 0 is introduced to account for the period when the plant is completely shut down, consequent of either PM or component failure. Transitions between the non-zero output levels are triggered by the EC, and are, therefore, considered forced transitions. The alternative path for gas lift, node 17, is activated on the unavailability of node 16 and deactivated when available. It, therefore, has a standby relationship with the latter and exists in two states, active (state 1) and standby (state 2). Nodes 1, 2, and 3 are affected accordingly on its activation or deactivation, as specified in Equation 4.5.

$$X_{gas} = \ell \left(\eta_6 - X_{lift} - \sum_{i \in \mathbf{w}} c_x^{\{i\}} \right) \qquad (4.6)$$
$$X_{oil} = \eta_{11}, \qquad X_{water} = \eta_{13}$$

The plant is separated into two subsystems, on the basis of commodity transported. The paths transporting gas, oil, and water are considered a single subsystem (production subsystem), by virtue of their independence. The flare is excess gas, and it is, therefore, discarded, since it has no effect on the gas output. The demands at nodes 7, 11, and 13 are respectively taken as the nominal Well output of gas, oil, and water. The capacities of nodes 8, 9, 14, and 15 are each 0.1×10^6 units of gas, and those of nodes 16 and 17, 1×10^6 units of gas. These nodes, according to the plant's schematic (Figure 4.13) appear to be competing with node 7 for the gas output from the TEG. In reality, the quantity of gas required to keep the TGs and TCs in operation and the gas lift are used up first, and any excess exported via node 7. This, however, is not considered by the subsystem's network model. Therefore, the gas output (flow through node 7), as deduced from the

network model should be corrected. The effective gas output is the difference between the gas flow into the TEG, the quantity used for gas lift ($X_{lift} = 1 \times 10^6$), and the gas consumed by any available TCs and TGs. Following flow calculation, the effective outputs, X_{gas}, X_{oil}, and X_{water}, of gas, oil, and water are given by Equation 4.6. Where, η_i is the flow through node i, $\mathbf{w} = \{8, 9, 14, 15\}$, and ℓ is an indicator function that takes the value 1 when $\eta_7 > 0$, and 0, otherwise.

Figure 4.14 shows the plant's network model, with the maximum flow along each link indicated. Flow along the gas production line is in Mega units, the oil and water lines, in kilo units, and the electricity line, in MW. The electricity network is considered a separate subsystem, as shown in Figure 4.14. Nodes 21 to 24 are demand points (local sources) for nodes 12, 10, 16, and 6 from the production subsystem. They, therefore, exist in two states, active, when their respective dependent nodes are working, and inactive, otherwise. Node 20 is a dummy node, assumed perfectly reliable, and assigned a constant capacity of 26 units, the combined maximum output of the TGs. The minimum threshold flows, Λ_{21} and Λ_{23}, of nodes 21 and 23 are set to 5.99 and 6.99 units, respectively, to account for the unavailability of one TG. With only one TG available, the flows through nodes 21 and 23 fall below their thresholds, and are shut down, as explained in Section 4.3.3. This augments the flows through nodes 22 and 24 to their required levels and keeps the EC and the oil pump in operation. The demands at the three output nodes are fixed, and the oil and water pumps, as well as node 20, are perfectly reliable. Their reconfiguration (shut down and restart), therefore, is unnecessary. This condition has been implemented by assigning a negative value to their threshold flows. With this manipulation, the shut-down requirement due to their effective load is never satisfied, since the actual load flows are non-negative. The remaining nodes are assigned a 0 minimum threshold flow requirement.

4.6.2.2 Production Level Determination

To determine the production availability of the plant, the evolution of X_{gas}, X_{oil}, and X_{water} are recorded as the simulation progresses. At the end of the simulation, the possible performance levels of each commodity are determined from the performance history of its relevant output node. The possible combinations of performance levels of the three commodities are generated, and their occurrences in the simulation history identified, to deduce the possible plant performance levels.

4.6.3 Simulation Results

A Matlab application was developed to model the plant under the following scenarios, CM only by one team (Case 1), CM only by two teams (Case 2), and both CM and PM by two teams; one dedicated to each maintenance type (Case 3). 1×10^5 Monte Carlo simulation samples of the plant's performance evolution for a mission time, $T_m =$

Table 4.3: Production levels of individual commodities.

Production Level	Commodity		
	Gas ($\times 10^6$)	**Oil** ($\times 10^3$)	**Water** ($\times 10^3$)
1	0	0	0
2	0.9	21.2	6.4
3	1	23.3	7
4	2.6		
5	2.7		
6	3		

Table 4.4: Gas production level probabilities.

Production Level	State Probabilities		
	Case 1	**Case 2**	**Case 3**
6	9.22×10^{-1}	9.30×10^{-1}	7.78×10^{-1}
5	3.85×10^{-2}	3.60×10^{-2}	8.95×10^{-2}
4	4.80×10^{-3}	4.70×10^{-3}	4.09×10^{-2}
3	2.50×10^{-3}	1.10×10^{-3}	5.90×10^{-3}
2	3.06×10^{-2}	2.76×10^{-2}	8.19×10^{-2}
1	1.90×10^{-3}	7.84×10^{-4}	3.80×10^{-3}

Table 4.5: Oil production level probabilities.

Production Level	State Probabilities		
	Case 1	**Case 2**	**Case 3**
3	9.52×10^{-1}	9.57×10^{-1}	8.58×10^{-1}
2	4.62×10^{-2}	4.19×10^{-2}	1.38×10^{-1}
1	1.90×10^{-3}	7.84×10^{-4}	3.84×10^{-3}

Table 4.6: Water production level probabilities.

Production Level	State Probabilities		
	Case 1	**Case 2**	**Case 3**
3	9.52×10^{-1}	9.57×10^{-1}	8.58×10^{-1}
2	5.20×10^{-3}	4.80×10^{-3}	4.26×10^{-2}
1	4.29×10^{-2}	3.79×10^{-2}	9.90×10^{-2}

Table 4.7: Plant production levels identified.

Output Type	Production Level						
	1	**2**	**3**	**4**	**5**	**6**	**7**
Gas ($\times 10^6$)	3	0.9	2.7	1	2.6	0.9	0
Oil ($\times 10^3$)	23.3	23.3	21.2	21.2	21.2	21.2	0
Water ($\times 10^3$)	7	7	0	0	6.4	6.4	0

1×10^3 hours, were studied in cases 1 and 2 while 3×10^4 samples for a mission time of 2×10^5 hours were studied in case 3. A much larger mission time was used in case 3 to accommodate several cycles of PM type 4, occurring once every 43800 hours.

Six production levels of gas and three each, of oil and water were identified by the simulation algorithm. These were ordered from lowest to highest and assigned production level numbers, as shown in Table 4.3. Their steady-state probabilities are given in

Table 4.8: Comparison of plant production level probabilities.

Production Level	State Probability		
	Case 1	Case 2	Case 3
1	9.22×10^{-1}	9.30×10^{-1}	7.74×10^{-1}
2	2.99×10^{-2}	2.74×10^{-2}	8.03×10^{-2}
3	3.84×10^{-2}	3.60×10^{-2}	8.93×10^{-2}
4	2.50×10^{-3}	1.10×10^{-3}	5.90×10^{-3}
5	4.70×10^{-3}	4.70×10^{-3}	4.09×10^{-2}
6	3.11×10^{-4}	1.43×10^{-4}	1.70×10^{-3}
7	1.90×10^{-3}	7.84×10^{-4}	3.80×10^{-3}

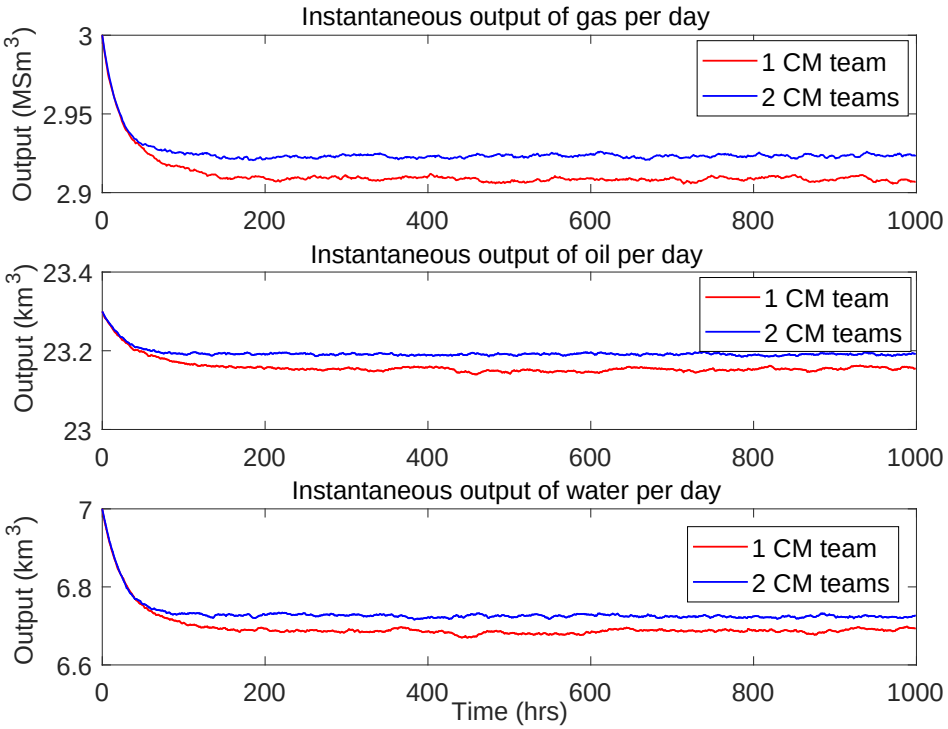


Figure 4.15: Expected instantaneous plant performance under CM only.

Tables 4.4-4.6. At the plant level, 7 production levels were identified, as presented in Table 4.7. The probabilities of the plant residing in any of these levels during a mission are presented in Table 4.8. Figure 4.15 shows the instantaneous production under CM only, for both 1 and 2 maintenance teams. As expected, the plant performs better with two maintenance teams. Overall, its availability at the nominal level improves, albeit slightly (see Table 4.8). However, both scenarios yield the same performance, within the first 30 to 45 hours of operation. This is explained by the high initial reliability of components, such that there are only a few failures, which can be conveniently covered by even a single maintenance team. As fatigue creeps in, failed components begin to queue, and more than one maintenance team is required. It is evident in Table 4.8 and

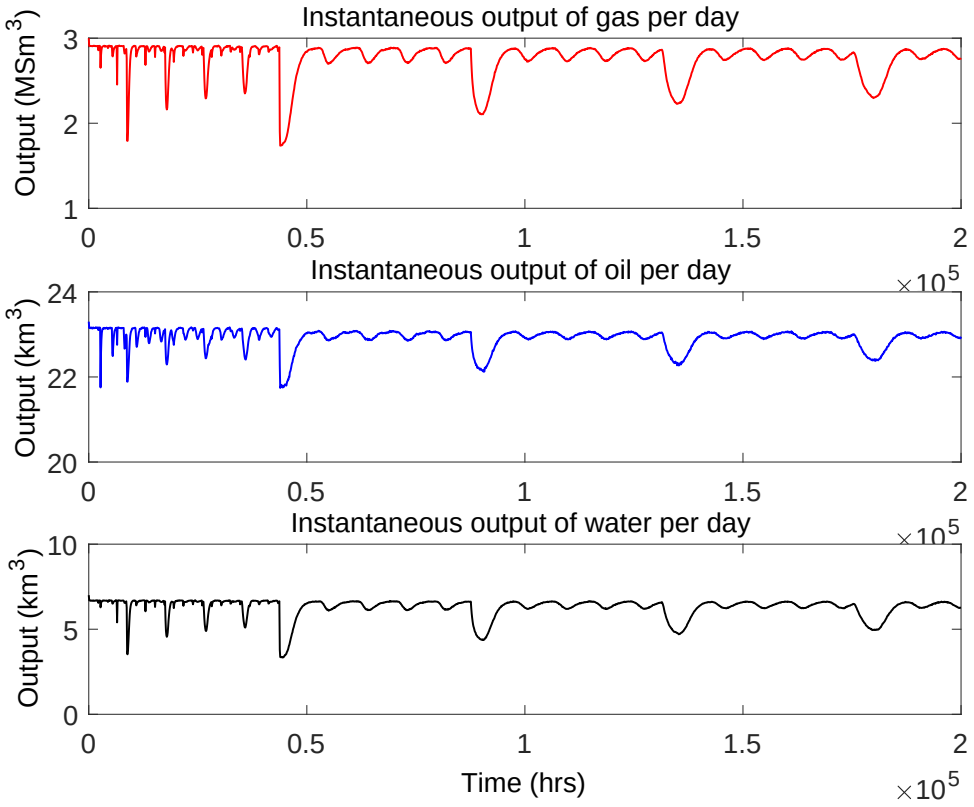
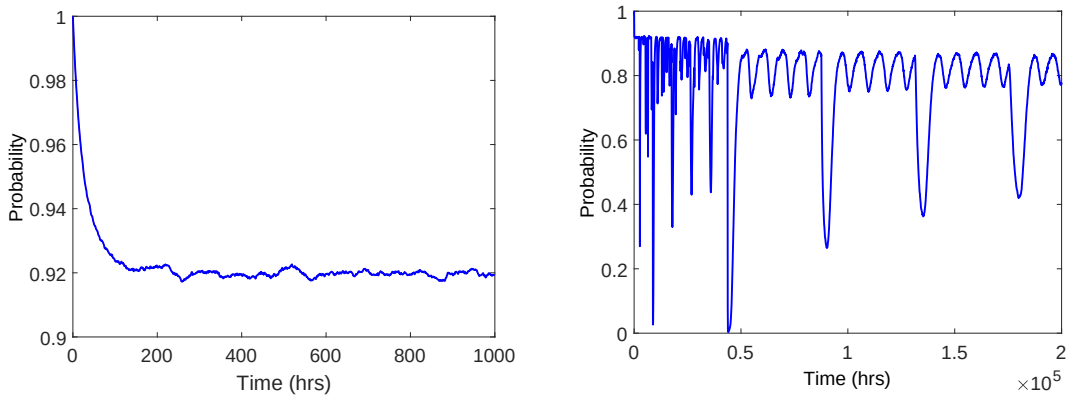


Figure 4.16: Expected instantaneous plant performance under CM and PM.



(a) CM only by 1 team

(b) CM and PM

Figure 4.17: Plant availability relative to state 1.

Tables 4.4-4.6 that the overall performance drops with PM. This is attributed to the fact that components exhibit exponential failure characteristics. PM increases their unavailability without improving their reliability [161]. Consequently, the smooth curves in Figure 4.15 are replaced by the rough curves in Figure 4.16, the deep drops in performance being due to the PM of critical components like the TEG. A similar trend is

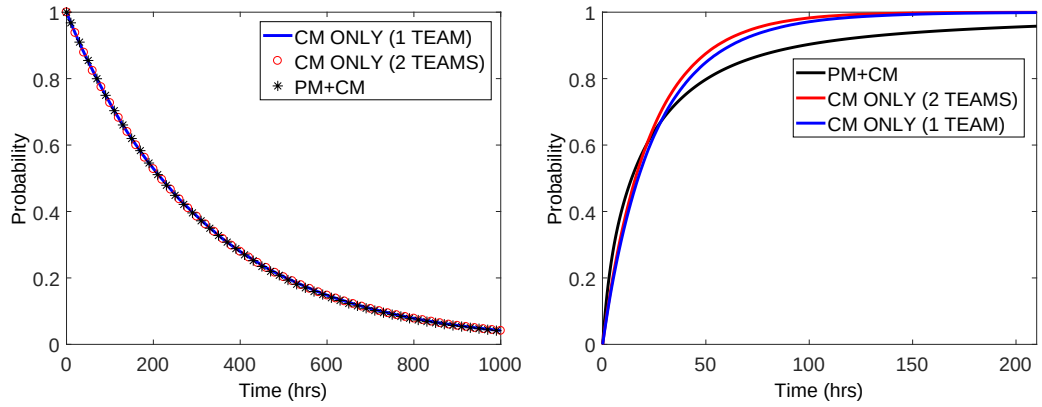
Table 4.9: Expected annual production.

Commodity	Expected Cumulative Output		
	Case 1	Case 2	Case 3
Gas	1.062×10^9	1.069×10^9	1.007×10^9
Oil	8.453×10^6	8.464×10^6	8.366×10^6
Water	2.446×10^6	2.456×10^6	2.292×10^6

Table 4.10: Expected annual production compared with Zio's result.

Commodity	Expected Cummulative Output					
	Case 1			Case 3		
	Proposed Approach	Zio's Approach	% Error	Proposed Approach	Zio's Approach	% Error
Gas	1.062×10^9	1.065×10^9	0.28	1.007×10^9	1.069×10^9	5.80
Oil	8.453×10^6	8.482×10^6	0.34	8.366×10^6	8.154×10^6	2.60
Water	2.446×10^6	2.447×10^6	0.04	2.292×10^6	2.446×10^6	6.30

portrayed by the plant's instantaneous availability, as shown in Figure 4.17.



(a) Reliability

(b) Recovery Probability

Figure 4.18: Plant reliability and recovery probability relative to state 1.

4.6.3.1 Expected Production

A frequently used indicator of performance is the expected cumulative amount of commodity flow through output nodes within a specified period. Using the data in Tables 4.4 to 4.6 and the identified production levels of each commodity, the expected annual outputs of gas, oil, and water are as presented in Table 4.9. These values are compared to those obtained by Zio et al. [161], as shown in Table 4.10. Only the results for cases 1 and 3 have been used here because case 2 was not considered by Zio et al.

4.6.3.2 Reliability and Recovery

The reliability and recovery of the plant are defined with respect to its nominal production level (state 1). Using the algorithm proposed in Section 4.5.1, the two quantities were obtained as shown in Figure 4.18. The structure and properties of the plant are

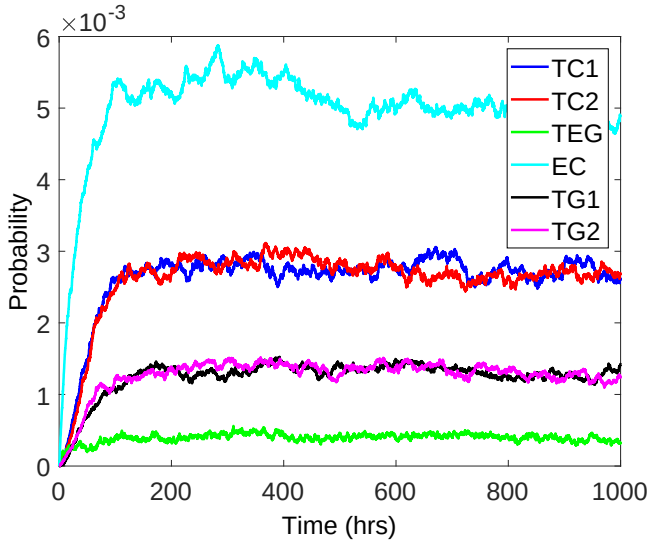


Figure 4.19: Maintenance response inadequacies for one CM team.

such that, the unavailability of any component leads to the plant's deviation from nominal performance. Since maintenance comes into play only after component failure, plant reliability is unaffected by the number of maintenance teams. Also, as outlined in Table 4.2, four types of PM actions are applicable but the earliest starts after 2160 hours. This implies, plant performance during the first 2160 hours is unaffected by PM. These considerations explain why the same reliability curve was obtained for all the three cases, as Figure 4.18(a) shows. Unsurprisingly, Figure 4.18(b) suggests 2 corrective maintenance teams ensure a higher recovery probability, explained by the increased response to component failures. For recovery within the first 22 hours of deviation from nominal performance, the policy implementing CM and PM gets the upper edge. This is because a significant proportion of these deviations is due to type-1 PM, with a mean duration of only 4 hours. In most instances, however, the PM of some component may be due while another component is under corrective maintenance. Given PM is only carried out when the plant is in its perfect state, the component's PM is deferred until all failed components are repaired. Even though the plant momentarily returns to nominal performance after the last repair, this is not regarded a recovery, as nominal performance is lost instantaneously when the queueing component is shut down for PM. This explains why the recovery probability for case 3 is on average the least.

4.6.3.3 Effects of Real-time component ranking

The plant's production availability was reassessed using real-time component ranking, and by maximizing gas production. Though the same outcome yielded by the predefined priority ranking shown in Table 4.1 was obtained, real-time ranking presented an intuitive alternative with little additional computational burden.

Table 4.11: Maximum gains from maintenance team scale-up.

Commodity	Expected Output	Percentage Gain	
		Case 1	Case 2
Gas	1.070×10^9	0.75	0.09
Oil	8.467×10^6	0.17	0.04
Water	2.458×10^6	0.49	0.08

4.6.3.4 Effects of limited maintenance teams

The plant's maintenance response inadequacy with respect to each of its six maintainable components was obtained. To enhance this, their state transitions during simulation were collected and saved as a function of time. As shown by their state-space diagram, TCs and TGs can be shut down from state 2 (see Fig. 4.12) but that does not remove them from the maintenance queue. Since maintenance response inadequacy defines how likely it is to have a component in the maintenance queue, transitions to and from shut-down (state 6) were not recorded. Figure 4.19 shows the maintenance response inadequacies of the plant under one corrective maintenance team.

To investigate the effects of limited maintenance on system performance and quantify the possible gains from a maintenance team scale-up, the plant was re-analysed with an unlimited number of maintenance teams. The expected annual production levels were obtained and compared with the values presented in Table 4.9. The expected output, with unlimited maintenance teams and the possible gains in cases 1 and 2 are given in Table 4.11. The table reveals, gas production is most effected by limited maintenance, and that case 2 is already close to the optimum number of maintenance teams.

4.6.4 Comments and Discussions

The proposed simulation and modelling approach has allowed us to obtain the same production levels identified via hand calculation by the original authors [161]. In addition, it yielded availability values at the nominal level that are within 6% of the reported values, even though 70% less samples were used in case 3. The expected output of the plant has been used as the reference parameter to assess the accuracy of the proposed approach because, it is normally the performance index of interest in multi-state system availability analysis. The 'noticeable' error in the results of Case 3 are attributable to the much smaller number of samples used in the proposed approach. Using 19 cores on a 1895.257MHz AMD Opteron(tm) 6168 processor, cases 1 and 2 took an average of 10.69 minutes and case 3, a few hours. Cases 1 and 2 required an additional 2.13 minutes when the plant was re-analysed using real-time component ranking.

To verify the effects of the modifications made to the flow calculation procedure, the case studies presented in Chapter 3 were re-analysed on the same computer. Prior calculation of node flows improved the simulation speed by 52.9% in case study 1 and

39.73% when applied to an unpublished 14-node system. However, it was not a feasible alternative when the 21-node system presented in the second case study was considered. The verification outcome suggests, the smaller the size of the matrix, $\boldsymbol{\beta}$, (see Section 4.4), the more advantageous the alternative of calculating and storing node flows prior to simulation. In addition, storage problems may be encountered with large systems, since node flows for all the possible system configurations have to be stored. These constraints make flow calculation during simulation inevitable for large systems, resulting in increased computational burden. The increased burden, however, can be mitigated with access to parallel computing, where the required number of simulation samples is shared across several computers or several workers on a multi-core computer.

The plant under consideration can exist in 437 configurations, considering corrective maintenance alone. Traditional approaches would require matching each of these to a production level [161]. Since this procedure can be time-consuming and error prone, Zio et al [161], proposed an innovative approach based on the minimal and maximum cut-sets of each production level. This approach, however, requires considerable human effort and a detailed knowledge of the plant's operational dynamics. It also suffers the setback of not being sufficiently general and intuitive, as a system's cut-sets and performance levels depend on its structure and the properties of its components. Therefore, every system would require a unique approach and a unique degree of difficulty.

Though the approach this chapter proposes is computationally more demanding than Zio et al's [161], it does not require the manual identification of production levels and enumeration of system cut sets. All it requires are the definition of inter-component relationships, component properties, and the structure of the system. The rest of the analysis is carried out by efficient algorithms. These attributes, coupled with the fact that it allows system structure to be defined by an adjacency matrix, make the proposed approach easily applicable to any system structure. Considering the time and human effort involved in the manual identification of production levels and the possibility of costly errors, the proposed approach is an efficient and credible alternative. Its advantages particularly stand out when applied to complex systems.

4.7 Chapter Summary

In this Chapter, an efficient and powerful simulation tool has been presented for the availability assessment of complex multi-state systems with interdependencies, multi-commodity flows, and limited maintenance teams. Algorithms for quantifying the relevant system availability and performance indices, including a new metric for the inadequacy of maintenance response have also been presented. The proposed simulation approach can implement reconfiguration requirements and derive system performance without reference to the system cut-sets or predefined system performance levels. Traditional approaches, however, would require the manual listing of all the system per-

formance levels and their associated cut-sets, which difficulty increases with system complexity and size. This attribute, therefore, is a key advantage and an illustration of its intuitiveness. Its applicability has been demonstrated by assessing the availability of a multi-commodity offshore plant operated by limited maintenance teams. By only defining the intra and inter component relationships, the approach provided (within an acceptable time frame) an outcome similar to one in literature, without prior knowledge of the plant's production levels or cut-sets. This renders it less dependent on human effort, intuitive, robust to human-induced errors, and suitable for any system architecture. It is implemented in the open-source uncertainty quantification tool, OpenCossan [109, 110], and, therefore, readily available to academics and industry.

Chapter 5

Probabilistic Risk Assessment of Station Blackout Accidents

5.1 Introduction

Adequate AC power is required for decay heat removal in nuclear power plants. Station blackout accidents, therefore, are a very critical phenomenon to their safety. Though designed to cope with these incidents, nuclear power plants can only do so for a limited time, without risking core damage and possible catastrophe. Their impact on a nuclear power plant's safety is determined by their frequency and duration, which quantities, currently, are computed via a static fault tree analysis that deteriorates in applicability with increasing system complexity. This Chapter proposes a novel alternative framework based on a hybrid of Monte Carlo methods, multi-state modelling, and network theory. It is, in other words, an adaptation of the frameworks in Chapters 3 and 4 to station blackout modelling. The intuitive framework is applicable to a variety of station blackout problems and can provide a complete insight into their risks. Most importantly, its underlying modelling principles are generic, and, therefore, applicable to non-nuclear system reliability problems, as well. When applied to the Maanshan nuclear power plant in Taiwan, the results which are also published in [54], validate the framework as a rational decision-support tool in the mitigation of station blackouts.

The remainder of this chapter is organised as follows; the next section reviews the proposed approach, highlighting its merits over the existing techniques. Section 5.3 provides a detailed description of the proposed station blackout (SBO) accident modelling framework. The simulation procedure, the computation of the relevant SBO indices, as well as their incorporation into the existing fault tree modelling formalism, are discussed in Section 5.4. Section 5.5 presents a practical case study, illustrating the application of the proposed modelling framework. Finally, a conclusion is drawn on the proposed framework in Section 5.6, with insights into its applicability and future development.

5.2 The Proposed Approach and Scope

As evidenced in Rao's, Rocha's, and Lei's works [72, 115, 120], Monte Carlo simulation (MCS) is flexible enough to model any system attribute. Its problem, however, is that most of the existing MCS algorithms are system-specific and require either the structure function, cut sets, or path sets of the system. An intuitive event-driven MCS procedure, offering multi-state component modelling opportunities was proposed in Chapter 3. This procedure is general and does not require the definition of the system's path & cut sets or structure function, thanks to its embedded graph model.

In this chapter, the graph and multi-state models proposed in Chapter 3 are adopted. The graph model is used to model the topology of the system and allow the performance of the system to be directly computed from the performance of the components. This attribute eliminates the need for an explicit association of component failure combinations to the state of the system. The multi-state model, on the other hand, is used to model the behaviour of the components, overcoming the assumption of a perfectly binary behaviour. It is particularly useful to the multiple failure mode and dynamic attribute representation of the Emergency Power Systems. This model, for instance, could be exploited to investigate the effects of limited maintenance teams (see Chapter 5, for instance) or the unavailability of spares on the Emergency Power Systems recovery [52]. The original model is extended to incorporate interdependencies by means of a dependency matrix and an efficient recursive algorithm to propagate the effects of failures across the system. Completing the framework, a simple MCS algorithm that induces LOOP in the system, replicate the ensuing sequence of events, and monitor the availability of power at the various safety buses, is proposed. The number of available safety buses, as a function of time, is computed after each system event. From the simulation history, any SBO index can be computed, thereby providing an opportunity for more insights into SBO risks. The multi-state component model, together with the dependency matrix, adequately captures and represents the redundancies in the emergency power system of the plant. Consequently, the explicit modelling of these redundancies, which sometimes poses a significant challenge, is eliminated.

5.2.1 Merits & Novelty of Proposed Approach

The framework proposed is limited to grid and switchyard induced LOOP, given their dominance [44]. Its preliminary results were first presented at the 13^{th} Probabilistic Safety Assessment and Management conference [53]. However, this chapter proposes several improvements. Firstly, an extensive review of the suitability of fault trees and their derivatives, to SBO analysis has been included. The effects of Common-Cause Failures (CCF), unavailability due to test or maintenance, and human error on the SBO frequency and recovery probability have also been considered. The chapter also shows how the results obtained from the framework can be incorporated into the existing

model. Finally, the number of computable SBO indices are extended and the effects of system configuration and sequence of operator response on system recovery, considered.

To the best of my knowledge, this chapter is the first documented application of load-flow simulation to a complete SBO risk assessment. With respect to the existing models discussed in Section 2.4.1, the proposed framework exhibits the following advantages.

- **Adequacy & Flexibility** - it models realistic attributes of the plant's power recovery and provides more insights into SBO risks. For instance, it enhances the investigation of the possibility of a second SBO after the first.
- **Convenience & Generality** - it is convenient in the sense that the modeller does not need to deduce the combination of component failure leading to system failure. They also do not need to explicitly model component redundancies, as these are implicitly captured by the modelling framework. In addition, the modelling framework is applicable to many system reliability problems.

5.2.2 Solution Sequence

The proposed approach is applied as summarised by the following chronological steps.

1. Identify the key elements of the system, define its topology, and derive its flow equation parameters.
2. Develop the multi-state model for each system element.
3. Model the interdependencies between the elements.
4. Force a LOOP event and simulate the behaviour of the standby power systems.
5. Compute the SBO indices from the simulation history.

5.3 Station Blackout Modelling

A nuclear power plant's power system consists of the grid, the switchyard, the emergency power systems, alternative emergency power systems, and the safety buses. The alternative emergency power systems are additional emergency sources (such as gas turbine generators) available at some plants to boost their LOOP/SBO recovery capability. In this section, it is shown how the plant's power system is accurately modelled and analysed, in line with the solution sequence outlined in Section 5.2.2.

5.3.1 The System Topology

The topology of the plant's power system is represented by a graph which nodes depict the components of the system. Connecting the nodes are perfectly reliable links portraying the direction of power flow. Flows from all the safety buses are terminated on a

virtual node introduced to represent the total available power. This virtual node would later be used to compute the non-recovery probability of AC power.

$$\boldsymbol{A} = \{a_{ij}\}_{M\times M} \mid a_{ij} = \begin{cases} 1 & \text{If flow is } i \rightarrow j \\ 0 & \text{Otherwise} \end{cases} \tag{5.1}$$

$$\boldsymbol{\Theta}\{X_{ij}\}_{k\times 1} \leqslant \{c_x^{\{i\}}\}_{M\times 1} \mid (i,j) \in \boldsymbol{e}, \quad \forall i \in \boldsymbol{V} \tag{5.2}$$

Let the nodes of the system be numbered from 1 to M and represented by the set $\mathbf{V} = \{1, 2, ..., M\}$, as proposed in Chapter 3. Since the links are perfectly reliable, the adjacency matrix, $\boldsymbol{A}$, of the system is as defined by Equation 5.1. The topology of the system, therefore, can be defined by $G \mid G = (\boldsymbol{V}, \boldsymbol{A})$. Using the parameters of G only, the flow equations of the system can be derived (see Chapter 3). These equations can then be used in synergy with the current state properties of the system nodes to deduce the performance of the system. For this, a linear programming algorithm is employed, given the possibility of flow redirection and the need to satisfy the capacity constraints of the nodes and their links. The objective is to find the flow across each link of the system that maximizes the flow into the virtual node. If X_{ij} is the flow across the link between nodes i and j and given there are k such links for all $(i,j) \in \boldsymbol{e}$, where $\boldsymbol{e}$ is the edge matrix of the system as defined in Chapter 3, the linear programming problem is formulated by Equations (5.2), (5.5), (5.7), and (5.8). Equation (5.2) expresses the inequality constraints to be satisfied, where $c_x^{\{i\}}$ denotes the capacity of node i when in state x. $\{c_x^{\{i\}}\}_{M\times 1}$, therefore, is the vector of current capacities of all the nodes of the system. The inequality matrix, $\boldsymbol{\Theta}$, is related to the incidence matrix, $\boldsymbol{\Gamma}$, as follows.

$$\boldsymbol{\Theta} = \{\theta_{iq}\}_{M\times k} \mid \theta_{iq} = \begin{cases} 1, & \gamma_{iq} \neq 0 \\ 0, & \text{otherwise} \end{cases} \tag{5.3}$$

$$\boldsymbol{\Gamma} = \{\gamma_{pq}\}_{M\times k} \mid \gamma_{pq} = \begin{cases} 1, & p = i \\ -1, & p = j \\ 0, & \text{otherwise} \end{cases} \tag{5.4}$$

$\boldsymbol{\Gamma}$ is related to $\boldsymbol{A}$ by (5.4), where $q = 1, 2, ..., k$ (the edge number) is the index of the edge between nodes i and j in $\boldsymbol{e}$ and $p = 1, 2, ..., M$. Equation (5.5) expresses the

$$\boldsymbol{\Phi}\{X_{ij}\}_{k\times 1} = \{0\}_{\eth\times 1} \quad \forall (i,j) \in \mathbf{e} \tag{5.5}$$

equality constraint to be satisfied, where $\boldsymbol{\Phi}$ and $\boldsymbol{\Gamma}$ are related as in Equation 5.6.

$$\begin{aligned} &\boldsymbol{\Phi} = \{\phi_{\lambda q}\}_{\eth\times k} \mid \phi_{\lambda q} = \gamma_{pq} \\ &\lambda = 1, 2, ..., \eth \mid \eth < M \quad f : \lambda \rightarrow p \quad \forall p \in (\mathbf{s} \cup \mathbf{t})' \end{aligned} \tag{5.6}$$

$$\begin{aligned} \mathbf{lb} &= \{0\}_{k\times 1}, \quad \mathbf{ub} = \{\Omega_{ij}\}_{k\times 1} \\ \Omega_{ij} &= \min\{c_{max}^{\{i\}}, c_{max}^{\{j\}}\} \quad \forall (i,j) \in \boldsymbol{e} \end{aligned} \tag{5.7}$$

$\eth$ is the number of intermediate nodes, $\boldsymbol{s}$, the set of source nodes, which comprises the grid and standby power systems, and $\boldsymbol{t}$, the virtual node denoting the total output. $\boldsymbol{\Phi}$, in effect, is a sub matrix of $\boldsymbol{\Gamma}$, containing all the rows of the latter corresponding to intermediate nodes. Equation (5.7) defines the lower and upper bound vectors, $\boldsymbol{lb}$ and $\boldsymbol{ub}$, of the flow through the links. Finally, the objective function of the linear programming problem is expressed in Equation (5.8).

$$\mathbb{O} = -\{\psi_q\}_{1\times k}\{X_{ij}\}_{k\times 1} \mid \psi_q = \sum_{i\in s} \gamma_{iq} \tag{5.8}$$

The total output of the system is given by the $\boldsymbol{t}^{th}$ element, $(\boldsymbol{\eta}, \mathbf{t})$, of $\boldsymbol{\eta}$. Interestingly, all the parameters, but $\{c_x^{\{i\}}\}_{M\times 1}$, required to compute $\boldsymbol{\eta}$ remain static during system simulation. The main task, therefore, is to update $\{c_x^{\{i\}}\}_{M\times 1}$ after each system event. The derivation of Equations (5.2) to (5.8) is available in Chapter 3.

5.3.2 The System Components

Each component is defined by a multi-state model that takes into account the various parameters that characterise its operation. Since the links have been assumed to be perfectly reliable, each component is defined as $\mathbb{E}_i = (\boldsymbol{T}, \boldsymbol{C}, x_0)$. $\boldsymbol{T}$ contains the density function objects for all the transitions depicted in the multi-state model of the component and $\boldsymbol{C}$ defines the capacity of the component in each state.

Each state capacity is expressed as a non-dimensional number defining the proportion of total system output the node can supply or transmit whilst residing in that state. If m is the total number of power trains at the plant, χ, the number of power trains the node simultaneously supplies, u, the proportion of power train demand it can satisfy, its capacity when working perfectly is, $\chi u m^{-1}$. This positive number expresses the total system output as a fraction of the number of power trains/safety buses present at the plant. On this note, the grid and switchyard nodes are each assigned unity capacity when available and 0, otherwise. The virtual output node has a fixed capacity of 1, and each safety bus, a fixed capacity of m^{-1}.

5.3.2.1 Modelling the Grid and Switchyard

The grid is modelled as a 2-state node; 'Working', when available and 'Failed', otherwise. Though grid failures are mostly random, they are modelled as forced transitions, since they already are incorporated in the LOOP frequency. Most often, plants tap their AC power from multiple offsite sources, and grid failure is defined as the failure of all of these sources. The repair of at least one of the failed sources, however, is sufficient to achieve grid recovery. For this reason, the transition from 'Failed' to 'Working' is defined

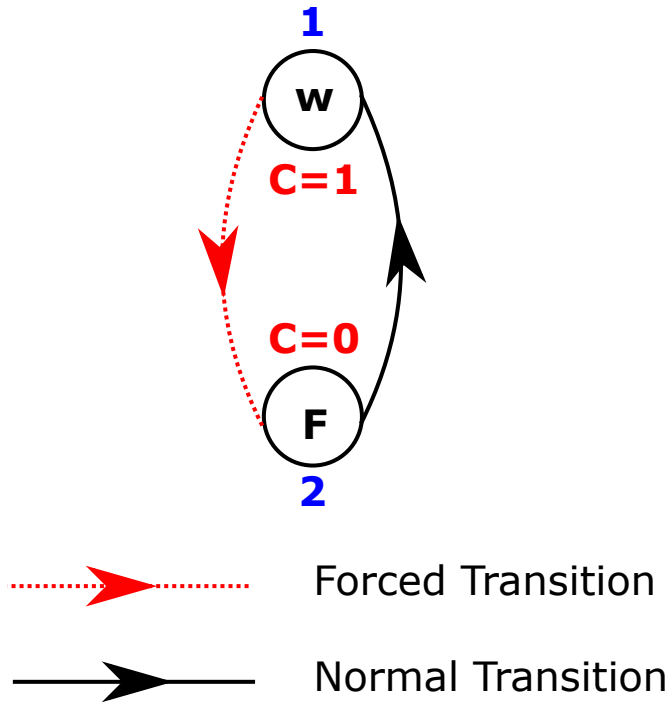


Figure 5.1: Multi-state model for Grid and Switchyard nodes.

by the upper bound of the envelope around the cumulative density functions (CDF) of the individual source repair distributions. Given this, sampling the grid recovery time entails generating a uniform random number and reading off its corresponding time from the envelope CDF, interpolating where necessary. An important point to note is, this approach slightly underestimates the grid recovery probability, as it assumes the individual source repair actions are initiated concurrently. In practice, the sources do not necessarily fail simultaneously and their recovery actions may commence at different times. This implies, by the time the last source fails, the restoration of already failed sources would have begun. The actual grid recovery time, therefore, is less than that given by the envelope CDF. This, however, is acceptable, as the goal in risk management is to ensure risk levels are acceptable, even in worst case scenarios.

Similarly, switchyard operation is defined by a 2-state node. If the plant is enhanced with multiple switchyards, switchyard recovery is treated as in the case of multiple grid sources. Figure 5.1 shows the multi-state model for the grid and switchyard.

5.3.2.2 Modelling the Standby Power Systems

The emergency power system is constituted by the emergency diesel generators, and in this work, gas turbine generators constitute the alternative emergency power system. In this section, only the multi-state behaviour of the standby power systems is modelled. The effects of redundancies on their operation is considered in a latter section. The following assumptions are invoked in developing these models.

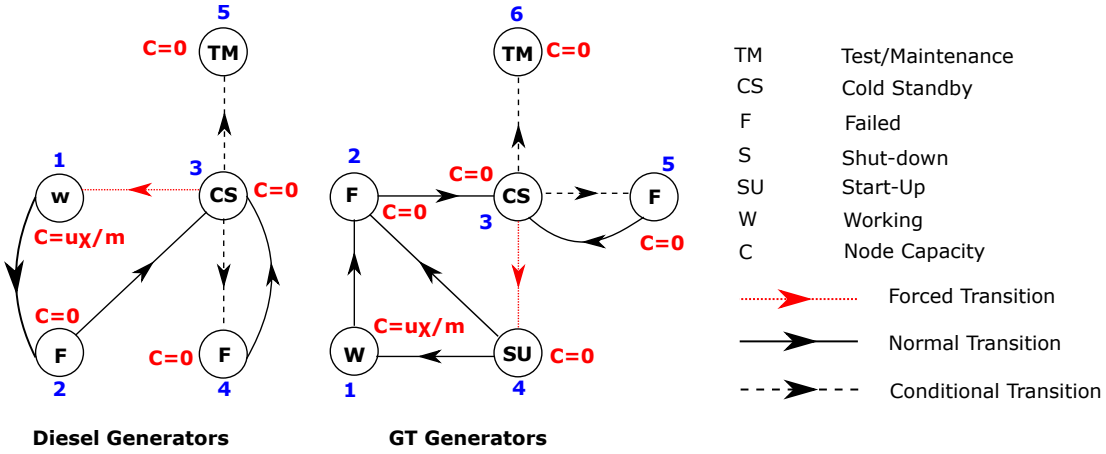


Figure 5.2: Models for emergency diesel and gas turbine generators without human error.

1. The initiation of test/maintenance is coincident with LOOP, and at any instance, there is not more than one source in test or maintenance.

2. Sources in test or maintenance remain unavailable through the sequence.

3. Repairs are commenced immediately.

4. A generator just from maintenance cannot fail to start, implying a perfect maintenance scenario.

The alternative emergency power System recovery is assumed offsite power recovery in [53]. This assumption is on the premise that their failure is included in the LOOP frequency. However, the assumption is impractical, given they are mostly a standby source. Their multi-state model, therefore, is modified to include running failures, rendering them an on-site source.

Failure-to-start and failure-to-run are considered the only failure modes an emergency diesel generator is susceptible to. Failure-to-start refers to the emergency diesel generator failure to start from cold-standby and failure-to-run denotes its failure to function for the duration of the LOOP. While the former is defined by a crisp probability, the latter is characterised by a time-to-failure probability density function. The Standardised Plant Analysis Risk (SPAR) model [43] considers a third emergency diesel generator failure mode, failure-to-load, defining the case when the emergency diesel generator starts but cannot power the load. This failure mode is considered failure-to-start, in the proposed framework. Two additional states, 'Working' and 'TM', are introduced as shown in Figure 5.2, to account for the perfect operation of the emergency diesel generator and its unavailability due to test or maintenance, respectively. Except otherwise, the transition from cold standby to working is instantaneous, whilst the transition from cold standby to failure or TM is also instantaneous but conditional. Conditional transitions are a special type of forced transition depending on a probabilistic event that

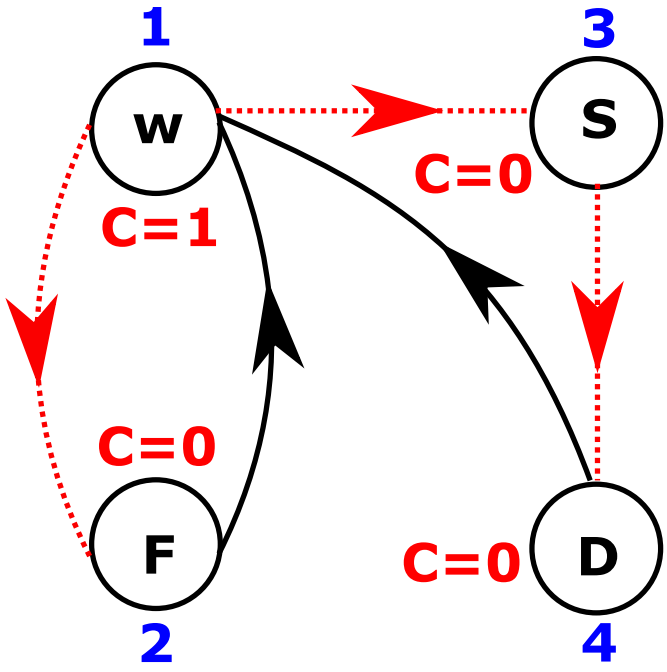


Figure 5.3: Multi-state model for switchyard with human error consideration.

is external to the component and with a known likelihood [52]. Conditional and forced transitions have the same representation in the transition matrix of the component.

The gas turbine generators behave in almost the same way as the emergency diesel generators, save for the difference in their start-up and manual alignment times. For this, a start-up state is inserted between their cold-standby and working states, as shown in Figure 5.2. Whilst in start-up, they could fail, explaining the transition $4 \rightarrow 2$.

5.3.2.3 Accounting for Human Error

Human error is very important in the risk assessment of engineering systems. In SBO recovery, human errors mostly manifest themselves as delayed response to a certain SBO mitigation action. For instance, the switchyard is forced into a temporary shut-down state during grid failures. On grid recovery, the plant personnel manually initiate its restoration, which process is susceptible to human-induced delays. Accounting for these delays, two additional states are introduced in the 2-state model discussed in Section 5.3.2.1, as shown in Figure 5.3. The transitions from 'Working' to 'Shut-down' and from 'Shut-down' to 'Delay' (D), are influenced by grid failure and recovery respectively. 'shut-down' denotes grid recovery-in-progress, while 'Delay' represents switching-in-progress. The latter determines the difference between the potential and actual bus recovery times. If this difference is negligible or the potential, instead of the actual bus recovery time is required, the model in Figure 5.1 is retained.

Similarly, the gas turbine generators and some emergency diesel generators require manual start-up and alignment, which is the case for shared diesel generators. A gen-

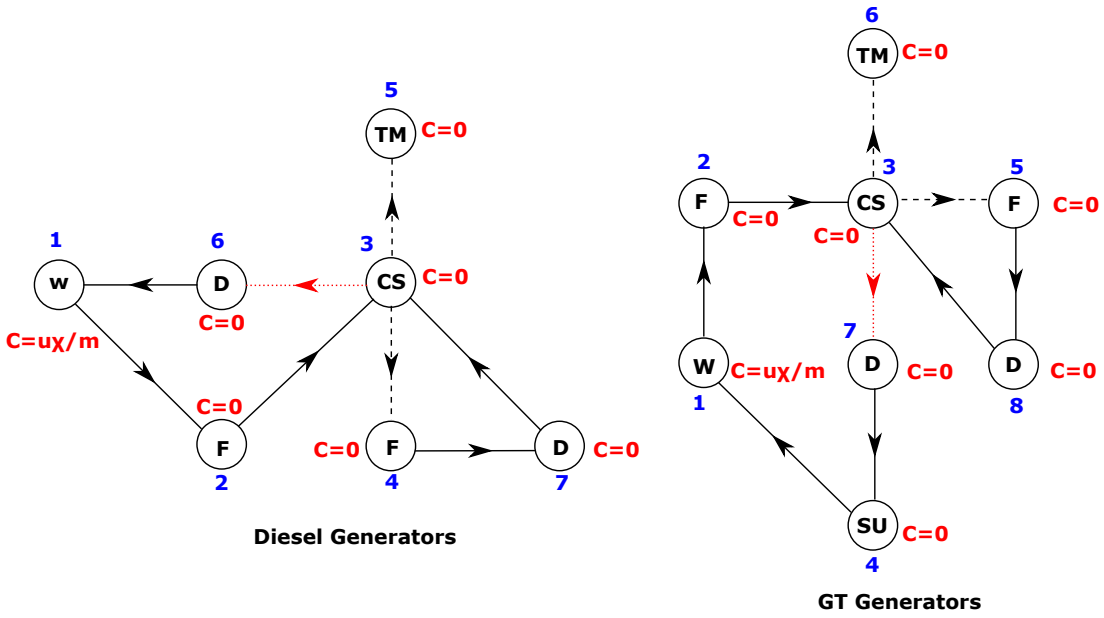


Figure 5.4: Models for emergency diesel and gas turbine generators with human error.

erator is said to be shared if it can substitute several units but, however, can only replace one unit at a given instance. Therefore, in the case of sequential multiple unit failures, only the first unit is replaced. For simultaneous failures, any of the units can be replaced, since they normally are identical. Since these replacements are manually executed, they are susceptible to delays, contrary to what most models suggest. Figure 5.2, for instance, assumes the transition from cold standby to the fully functional or failure state to be instantaneous. This, by extension, implies, any maintenance action (if the generator fails to start) is initiated at once. However, with human error, the start-up procedure may be initiated later than scheduled. Two additional states, therefore, are introduced, one each, between cold standby & working and failure & cold standby, as shown in Figure 5.4, to account for these delays. The plant personnel have been assumed to be well trained, experienced, and fit to perform their assigned tasks as expected. Consequently, the possibility of inappropriately executed actions is ignored.

Transitions $6 \rightarrow 1$ with $4 \rightarrow 7$ and transition $7 \rightarrow 4$ with $5 \rightarrow 8$, of Figure 5.4, account for human error in the recovery of manually operated emergency diesel and gas turbine generators, respectively. In practical applications, human error is expressed in terms of the probability of not completing a given action within a specified time. If this probability is known for multiple times, a CDF could be fitted through the points. For this, the Weibull distribution is recommended, since it can yield a wide range of distributions. Recall the CDF of a Weibull distribution is $1 - e^{-(t/a)^b}$, where a and b are its scale and shape parameters, respectively. Given the human error probabilities are the likelihoods of inaction, they define the complement of the human reaction time CDF. Therefore, the Weibull parameters, a and b, are obtained by fitting the set of

probability values to the function $e^{-(t/a)^b}$.

5.3.3 Modelling Component Interdependencies

To ensure resilience, system designers often employ multiple layers of defence, either in the form of redundancies or shared components. This proactive strategy inadvertently introduces interdependencies in the system, resulting in modelling accuracy issues. Interdependency is defined in a more general sense as the potential for a state change in one element to trigger a state change in another. Two models, the CCF and the cascading failure models, are proposed to implement these interdependencies.

5.3.3.1 The CCF Model

This model is used when the random failure of any member of a group of similar components, performing the same task could cause the failure of one or more of the remaining components [104]. Following from Section 2.2.1.1,

- there is a set of probabilities associated with the number of components involved in any random failure event. Let this set of probabilities, for group k, be defined by $\boldsymbol{\theta}^{\{k\}} \mid \boldsymbol{\theta}^{\{k\}} = \{\theta_r^{\{k\}}\}^{M_k}$, where r is the number of components involved in the failure event, M_k, the total number of components in the group, and $\sum_{r=1}^{M_k} \theta_r^{\{k\}} = 1$.
- all the components in a CCG fail in the same mode. Implying, the CCG for start-up failures cannot influence the CCG for running failures, for instance.

Each CCG, therefore, can be defined by the quadruple, $\left(\boldsymbol{\rho}^{\{k\}}, \beta_1^{\{k\}}, \beta_2^{\{k\}}, \boldsymbol{\theta}^{\{k\}}\right)$. Where, $\boldsymbol{\rho}^{\{k\}}$ is the set of components in the CCG, $\beta_1^{\{k\}}$, the common failure mode, and $\beta_2^{\{k\}}$, the state the components have to be in to be susceptible to this failure mode. The algorithm for propagating CCF is summarised as follows.

1. When a component fails, determine k and check if its new state matches $\beta_1^{\{k\}}$.
2. Go to step 5 if there is no match. Else, determine the number of components, r, that will fail.
3. Go to step 5 if $r = 1$. Else, remove from $\boldsymbol{\rho}^{\{k\}}$, the component initiating the failure event. From the remainder, randomly select $r - 1$ components.
4. For each component selected in step 3, check if its current state matches $\beta_2^{\{k\}}$ and set this to $\beta_1^{\{k\}}$.
5. End procedure.

The procedure above requires $\boldsymbol{\theta}^{\{k\}}$ to be in conformity with the α-factor model [104]. CCF probabilities expressed in other models would need to be converted as in [104].

5.3.3.2 The Cascading Failure Model

This model is used for interdependencies not satisfying the CCF criteria. For instance, the redundancies among the standby power systems and the dependence of the latter on the grid and switchyard. An important assumption invoked in this model, however, is that on occurrence of the trigger event, the dependent event occurs immediately.

Initially proposed in Chapter 4, the model defines interdependencies by a dependency matrix. The dependency matrix, $\boldsymbol{D}_i$, for node i, defines the effects of the node's state transition on other nodes. It takes the form, $\boldsymbol{D}_i = \{d_{j1}, d_{j2}, d_{j3}, d_{j4}\}_{v\times 4} \mid j = 1, 2, ..., v-1, v$, where d_{j1} is the state of i triggering the event, d_{j2}, the affected node, d_{j3}, the state the node has to be in to be vulnerable, and d_{j4}, its target state after the event. Each row of $\boldsymbol{D}_i$ defines the behaviour of an affected node, and v, the number of relationships. For example, consider a 2-component system, with each component existing in 3 possible distinct states. When component 1 makes a transition to state 3, component 2 is forced to make a transition to state 2 as well, if and only if the latter is currently residing in state 1. Since component 1 is the trigger, the interdependency is defined by $\boldsymbol{D}_1$ as,

$$\boldsymbol{D}_1 = \begin{pmatrix} 3 & 2 & 1 & 2 \end{pmatrix} \tag{5.9}$$

Let a third 3-state component be added to the system. In addition to its effect on component 2, let the transition of component 1 also affect component 3, such that the latter is forced to state 1 if it is in state 3 at the time of the trigger event. To represent the overall behaviour of component 1, $\boldsymbol{D}_1$ is updated as shown in Equation 5.10, which indicates that each row of the dependency matrix represents a possible outcome.

$$\boldsymbol{D}_1 = \begin{pmatrix} 3 & 2 & 1 & 2 \\ 3 & 3 & 3 & 1 \end{pmatrix} \tag{5.10}$$

Occasionally, a state change in a node can only affect another node if a third node is in a certain state. This type of dependency is known as a joint dependency, and it is outside the scope of the initial model in Chapter 4. The joint dependency matrix, $\boldsymbol{D'} = \{d'_{j1}, d'_{j2}, d'_{j3}, d'_{j4}\}_{v\times 4}$, is introduced (in this chapter) to resolve this problem. Element d'_{j1} defines the state the third node must be in to satisfy the joint dependency while d'_{j2}, d'_{j3}, and d'_{j4} have the same meaning as d_{j2}, d_{j3}, and d_{j4} respectively. Assuming a certain state change in node i only affects, say node x, if node ω is in state σ, $\boldsymbol{D}_i$ defines the relationship between nodes i and ω, while $\boldsymbol{D}'_\omega$ defines the relationship between ω and x. Nodes i, ω, and x are the trigger, intermediate, and target nodes, respectively. The intermediate node does not undergo a state change, meaning its target state is the same as its vulnerable state. Therefore, in $\boldsymbol{D}_i$, the 3^{rd} and 4^{th} elements of the row corresponding to the intermediate node are equal. Given $j = 1$, for $\boldsymbol{D}_i$, $d_{12} = \omega$, $d_{13} = d_{14} = \sigma$ and for $\boldsymbol{D}'_\omega$, $d'_{11} = \sigma$, $d'_{12} = x$. The remaining elements retain their

meaning, as defined earlier.

$$\mathbf{D}_1 = \begin{pmatrix} 3 & 2 & 1 & 2 \\ 3 & 2 & 2 & 2 \end{pmatrix} \quad \mathbf{D}'_2 = \begin{pmatrix} 2 & 3 & 3 & 1 \end{pmatrix} \tag{5.11}$$

Let, for illustrative purposes, the dependency between components 1 and 3 (second row of $\boldsymbol{D}_1$ in Equation 5.10 only hold if component 2 is in state 2. To represent this attribute, the second row of $\boldsymbol{D}_1$ is modified to reflect the relationship between components 1 and 2, and the relationship between components 2 and 3, defined by $\boldsymbol{D}'_2$, as shown in Equation 5.11. Notice $\boldsymbol{D}'_2$, instead of $\boldsymbol{D}_2$, has been used, since the relationship between components 2 and 3 is due to a joint dependency.

$$\boldsymbol{D}_1 = \begin{pmatrix} 3 & 2 & 1 & 2 \\ 3 & 2 & -3 & -3 \end{pmatrix} \quad \boldsymbol{D}'_2 = \begin{pmatrix} -3 & 3 & 3 & 1 \end{pmatrix} \tag{5.12}$$

The dependency and joint dependency matrices, indeed, can be used to represent a wide range of dependencies. However, there are instances that may result in large matrices, which cases require an intuitive manipulation, to keep the matrix size moderate and prevent errors. A negative sign is introduced in front of the trigger or vulnerable state to signify that the dependency is satisfied only if the component is **not** in that state. This notation is analogous to the **NOT-gate** in fault trees. For instance, if component 1, in the scenario above, affects component 3 only if component 2 is in states 2 or 1, it is efficient to exploit the **NOT** notation, instead of inserting an additional row in each of $\boldsymbol{D}_1$ and $\boldsymbol{D}'_2$. Recalling that component 2 has 3 states, state 2 **OR** state 1 is logically equivalent to **NOT** state 3. Hence, $\boldsymbol{D}_1$ and $\boldsymbol{D}'_2$ are as given in Equation 3.14.

A recursive algorithm is proposed to implement the dependency matrices. If x_i denotes the new/current state of node i, the algorithm is summarised as follows.

1. Define a register, $\boldsymbol{R}$, to hold the affected components, their vulnerable, and target states.

2. Using $\boldsymbol{D}_i$ and x_i , find all components affected by the state change and update $\boldsymbol{R}$ with elements 2 to 4 of the rows representing the components.

3. Select the last row of $\boldsymbol{R}$ and check if its last two elements are equal. This row defines the dependency induced in component ω by component i.

4. If the response to the query in step 3 is in the affirmative, designate the equal elements, ϵ, delete the last row of $\boldsymbol{R}$, and,

   (a) using ω, $\boldsymbol{D}'_\omega$, and x_ω as inputs, call steps 1 to 7, noting that a row in $\boldsymbol{D}'_\omega$ is affected by the state change only if its first element is ϵ.

   (b) Continue from step 3.

Else, proceed to step 5.

5. Force the designated transition as determined in step 3 and delete the last row of $\boldsymbol{R}$. If the affected node is in standby, and its target state, Working, Delay, or Start-Up, initiate its start-up procedure.

6. If $\boldsymbol{D}_\omega$ exists, repeat steps 2 to 6, replacing $\boldsymbol{D}_i$ and x_i with $\boldsymbol{D}_\omega$ and x_ω respectively.

7. Repeat steps 3 to 6 until $\boldsymbol{R}$ is empty, and terminate the procedure.

5.4 System Simulation & Analysis

The system's operation is imitated by generating random failure events of components and their corresponding repairs. For every component transition, the capacity vector, $\{c_x^{\{i\}}\}_{M\times 1}$, of the system is updated and used to deduce the flow, $(\boldsymbol{\eta}, \boldsymbol{t})$, through the output node. At time $t = 0$, the grid and switchyard nodes are in operation, while the emergency power systems and alternative emergency power systems are in cold standby. LOOP is initiated by setting the grid (for grid centred LOOP) or the switchyard to its failure state. The next transition parameters of the standby systems are sampled, and the simulation is moved to the earliest transition time, t. Components with next transition time equal to t are identified, the required transitions effected, their next transition times sampled, the new system performance computed, and the next simulation time determined. This cycle of events continues until offsite power is recovered.

Let $\boldsymbol{\mu}_{old}$ hold the node capacities at the previous system transition, $\boldsymbol{\tau}$, the vector of next node transition times, N, the number of simulation samples, and $\boldsymbol{S} = \{s_j\}^N$, the register indicating the occurrence of an SBO. The indicator register, $\boldsymbol{S}$, is such that, $s_j = 1$ if an SBO occurs in the j^{th} sample, and 0, otherwise. The simulation algorithm is summarised as follows.

1. Initialize the register storing the flow through the output node, set $N = 1$, $\mathbf{S} = \{\}$, and define the simulation stopping criterion. The stopping criterion could be the number of LOOP, number of SBO, or convergence of the SBO probability.

2. Determine which component will be unavailable due to test or maintenance.

3. Set $s_N = 0$ and $\boldsymbol{\tau} = \{\infty\}^M$, where M is the number of nodes in the system.

4. Force LOOP, as described earlier, accounting for interdependencies according to the procedures described in Sections 5.3.3.1 and 5.3.3.2. Remember to sample the next transition parameters after every node transition and update $\boldsymbol{\tau}$. See Chapter 3 for the procedure for sampling the transition parameters of a multi-state node.

5. Define $\boldsymbol{\mu}$ using the current states of the nodes, that is, $\boldsymbol{\mu} = \{c_{x_0}^{\{i\}}\}_{M\times 1}$ and set $t = 0$, $\boldsymbol{\mu}_{old} = \boldsymbol{\mu}$.

6. Determine $X_{out} \mid X_{out} = (\boldsymbol{\eta}, \boldsymbol{t})$ and save as a function of time.

7. Set $s_N = s_N + 1$ if $X_{out} = 0$ and determine the next simulation time, $t = min(\boldsymbol{\tau})$.

8. Find nodes with next transition time equal to t. For each node, force the required transition, sample its next transition parameters (except for nodes returning to cold standby), and update $\boldsymbol{\mu}$ & $\boldsymbol{\tau}$.

9. Restart nodes returning from repairs if $X_{out} < 1$.

10. If $\boldsymbol{\mu}_{old} \neq \boldsymbol{\mu}$,

    (a) compute X_{out} and set $s_N = s_N + 1$ if $X_{out} = 0$.

    (b) save X_{out} if different from the previous.

    (c) temporarily set the capacity of the switchyard node to 1 if it is in shut-down and calculate the new system flow. If this flow is non-zero, set the switchyard to start-up, sample its next transition parameters, and update $\boldsymbol{\tau}$.

11. Set $\boldsymbol{\mu}_{old} = \boldsymbol{\mu}$, $t = min(\boldsymbol{\tau})$, and check if offsite power is recovered.

12. Repeat steps 8 to 11 until offsite power is recovered. Discard history N if $s_N = 0$ and set $N = N + 1$.

13. Repeat steps 2 to 12 until the simulation stopping criterion is met, and terminate algorithm.

14. Compute the relevant SBO indices

5.4.1 SBO Indices: Computation & Relevance

The SBO frequency, f_s, makes the list of the most informative and desired SBO indices. It defines the expected number of times, per year, an SBO occurs at a plant. If $p_1^{\{sbo\}}$ defines the conditional probability of an SBO given a LOOP occurring at frequency, f_l,

$$\begin{aligned} f_s &= f_l p_1^{\{sbo\}} \\ p_1^{\{sbo\}} &= \frac{\sum(\mathbf{S} > 0)}{N - 1} \end{aligned} \tag{5.13}$$

per year, Equation 5.13 shows how f_s and $p_1^{\{sbo\}}$ are obtained from the system simulation history. The fraction of f_s occurring at start-up is deduced from the number of SBO at time 0. This index could be used to assess the efficiency of the start-up procedure, as well as the vulnerability of the on-site backup generators in cold standby.

The non-recovery probability, $r'(t)$, defines the likelihood of recovery duration from an SBO accident exceeding a given time. It is computed as detailed in Chapter 4, and

LOOP	ONSITE POWER FAILURE	REACTOR PROTECTION SYSTEM	RCS	AFW	EMERGENCY PRESURIZATI	RCP SEAL STAGE 1 INTEGRITY	RCP SEAL STAGE 1 INTEGRITY	RCP SEAL STAGE 2 INTEGRITY	RCP SEAL STAGE 2 INTEGRITY	OFFSITE POWER RECOVERY	ONSITE POWER RECOVERY
T(PG)	EM	K	Q	L(T)	X(E)	BP1	O1	BP2	O2	ER1	ER2

Figure 5.5: An excerpt from the SBO event tree showing headings.

like $p_1^{\{sbo\}}$, belongs to the set of inputs to the SBO event tree. Given it defines the unavailability of power at the plant, $r'(t)$ can be directly compared with the reliability of the SBO mitigating mechanism. The outcome of such a comparison would help ascertain the adequacy of the mitigating mechanism. In addition, $f_s \times r'(t)$ yields the frequency of exceedance, a measure of the overall SBO risk at the plant. It also presents a means of assessing the relative effectiveness of multiple recovery responses.

Finally, the conditional probability of a second SBO, $p_2^{\{sbo\}}$, the first is given by,

$$p_2^{\{sbo\}} = \frac{\sum(\mathbf{S} > 1)}{\sum(\mathbf{S} > 0)} \tag{5.14}$$

Knowledge of $p_2^{\{sbo\}}$ may shape the recovery response on the occurrence of a second SBO. For instance, a plant with a large $p_2^{\{sbo\}}$ would require the logistics used in the recovery of the first SBO left in the field and the operations staff kept on high alert. This reduces human error, ensuring a lower non-recovery probability, $r_2'(t)$, of the second SBO.

$$p_n^{\{sbo\}} = \frac{\sum(\mathbf{S} > n-1)}{\sum(\mathbf{S} > n-2)} \tag{5.15}$$

Generally, the conditional probability, $p_n^{\{sbo\}}$, of the n^{th} SBO given the $(n-1)^{th}$ SBO is expressed as in Equation 5.15. If, however, absolute probabilities are required instead, the denominator of the right-hand side of the equation is replaced with $N-1$.

5.4.2 Incorporation into the Existing Framework

Shown in Figure 5.5 is an excerpt from the SBO event tree presented in [43]. Of its 12 headings, only four; T(PG), EM, ER1, and ER2 are of relevance to SBO recovery. The first depicts LOOP, and requires the LOOP frequency. The second represents SBO occurrence, and requires the unavailability of the standby power systems. Here, the chain of complicated fault trees in the existing model can be replaced with the conditional SBO probability, $p_1^{\{sbo\}}$. The last two headings represent offsite and standby power recovery, respectively. These can be merged into one heading, say AC power recovery, and the complicated fault trees replaced with a crisp value read from $r'(t)$. With these, the core damage frequency induced by the first SBO is computed by solving

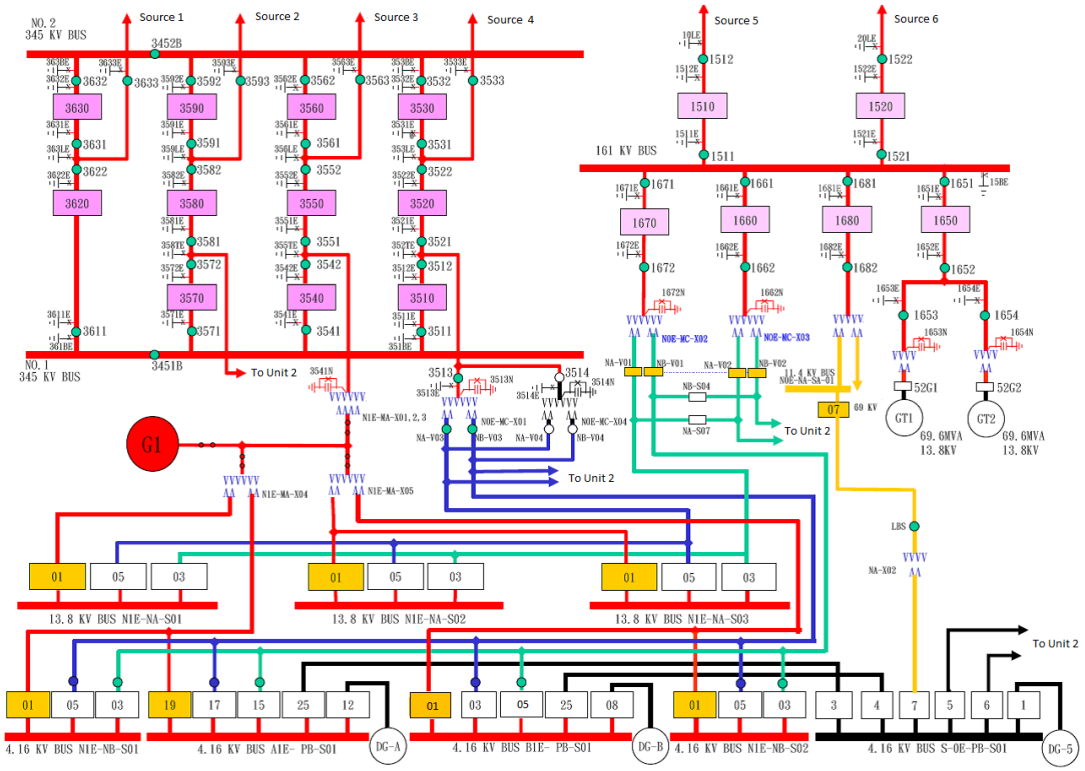


Figure 5.6: Layout of the Maanshan nuclear power plant AC distribution system.

the event tree using standard procedure. For the second SBO, the first is regarded the initiating event. The LOOP frequency, therefore, is replaced with f_s, $p_1^{\{sbo\}}$ with $p_2^{\{sbo\}}$, and $r'(t)$ with $r'_2(t)$.

5.5 Case Study:The Maanshan Nuclear Power Plant

Maanshan is a two-unit, 1902 MW, Westinghouse PWR nuclear power plant operated by the Taiwan Power Company. Its offsite power is supplied by six independent sources, four of which are connected to the 345 kV switchyard and the remainder, through the 161 kV switchyard. It is powered through two safety buses, AIE-PB-S01 and BIE-PB-S01, each with a dedicated emergency diesel generator; DG-A and DG-B, respectively. A shared emergency diesel generator, DG-5, connected as shown in Figure 5.6 is available as backup in case any of the dedicated generators is unavailable. In addition to the shared emergency diesel generator, are two gas turbine generators, GT1 and GT2, connected via the 161kV switchyard. These generators form the alternative emergency power system of the plant, each satisfying the demand on both power trains.

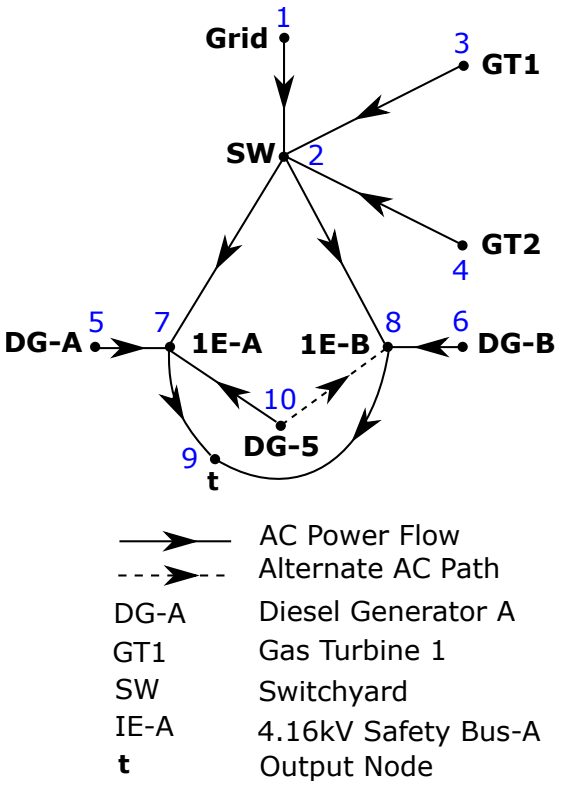


Figure 5.7: Simplified schematic of plant's AC distribution system.

During normal plant operation, the safety buses are fed by the main plant generator, G1, via the black lines and the normally closed breakers 19 & 01. On plant shut-down, G1 becomes unavailable, and the safety buses are forced to tap power from the 345kV switchyard (via the blue lines and the normally open breakers 17 & 03) or the 161kV switchyard (via the black lines and the normally open breakers 15 & 05). When these sources are also unavailable, DG-A and DG-B are automatically started and aligned. DG-5 is manually started and aligned by operators on the failure of any of these. The manual start-up and alignment procedure of GT1 and GT2 is initiated when at least 2 out of the 3 emergency diesel generators become unavailable. Following their successful start-up, the gas turbine generators take about 30 minutes attain full functionality.

An assessment of the plant's SBO risk due to grid and switchyard LOOP is required.

5.5.1 Developing the System and Component Models

Figure 5.7 is the simplified schematic of the plant's AC power system, showing all the elements relevant to an SBO. DG-5, though serving only one bus at a time, is assumed connected to both buses in the system's adjacency matrix. This implies, its flow is divided between the buses, contrary to what obtains in reality. However, since the flows from the two buses are emptied into the virtual output node, $\boldsymbol{t}$, the total flow from the

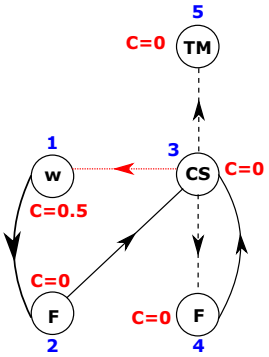


Figure 5.8: Multi-state model for the main diesel generators (DG-A & DG-B).

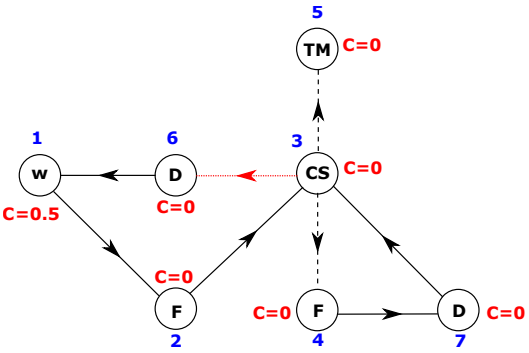


Figure 5.9: Multi-state model for the shared diesel generator (DG-5).

shared generator is accounted for. As shown, the six grid sources and the two switchyard sources have each been represented by single nodes, as proposed in Section 5.3.2.1.

Nodes 1, 7, 8, and 9 are modelled as proposed in Sections 5.3.2 and 5.3.2.1. The switchyard, on the other hand, is modelled according to Figure 5.3, to account for human error during its start-up from shut-down. Since DG-A (node 5) and DG-B (node 6) are automatically started following a LOOP, they are not susceptible to human error, and, therefore are modelled as shown in Figure 5.8. DG-5, GT1, and GT2, however, require human intervention for their start-up and alignment. Node 10, therefore, is modelled according to Figure 5.9 and nodes 3 and 4, according to Figure 5.10.

$$\boldsymbol{A} = \begin{matrix} (1,2) & 1 \\ (2,7) & 1 \\ (2,8) & 1 \\ (3,1) & 1 \\ (4,1) & 1 \\ (5,7) & 1 \\ (6,8) & 1 \\ (7,9) & 1 \\ (8,9) & 1 \\ (10,7) & 1 \\ (10,8) & 1 \end{matrix} \tag{5.16}$$

Justifying the values assigned to the state capacities of the generators, recall the system consists of 2 safety buses ($m = 2$), with each of DG-A and DG-B serving only one ($\chi = 1$). Since these generators can, however, fully meet the demand on the bus they serve ($u = 1$), they are assigned a capacity of 0.5 when working, as proposed in Section 5.3.2. The gas turbine generators, on the other hand, can fully serve both buses simultaneously ($\chi = 2$), and therefore, have a capacity of 1 when working. From the multi-state models, the capacity vector for the main diesel generators, the shared

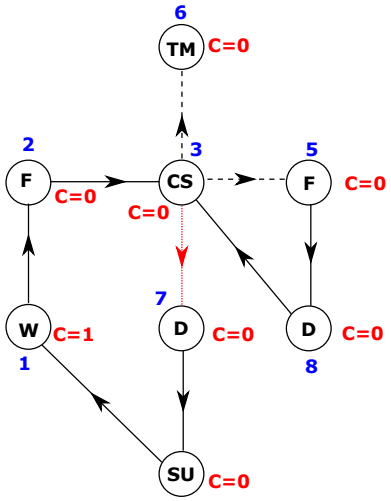


Figure 5.10: Multi-state model for the gas turbine generators (GT1 & GT2).

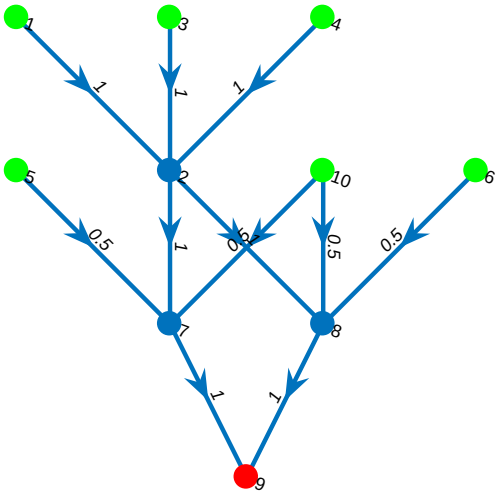


Figure 5.11: Full system graph model showing maximum flow along links.

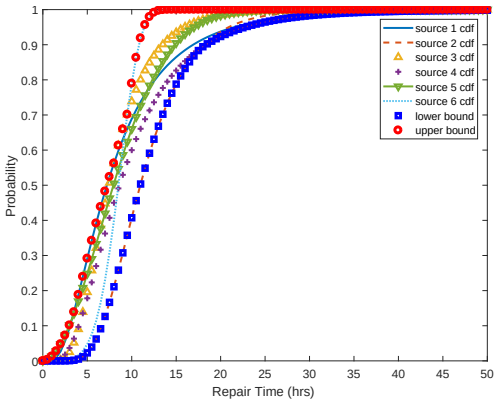


Figure 5.12: Effective repair CDF for multiple grid sources.

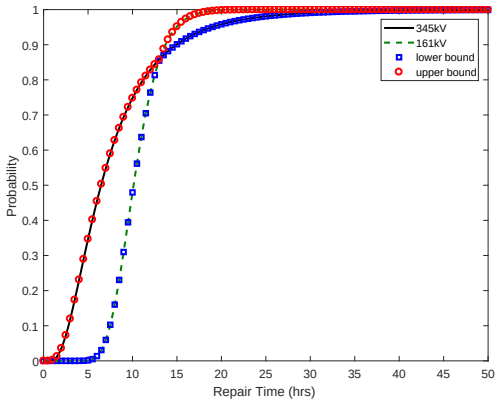


Figure 5.13: Effective repair CDF for multiple switchyard sources.

diesel generator, and the gas turbine generators are $\{0.5, 0, 0, 0, 0\}$, $\{0.5, 0, 0, 0, 0, 0, 0\}$, and $\{1, 0, 0, 0, 0, 0, 0, 0\}$, respectively. Using these parameters in conjunction with Figure 5.7, the adjacency matrix of the plant, expressed as a sparse matrix, is given in Equation 5.16. Given the adjacency matrix, the other parameters of the system flow equations are obtained as described in Section 5.3.1, where $\boldsymbol{s} = \{1, 3, 4, 5, 6, 10\}$ and $\boldsymbol{t} = 9$. Figure 5.11 is the system's graph model showing the maximum flow along each link, derived from the adjacency matrix and the maximum node capacities.

5.5.2 Component Reliability Data

Though realistic, the data used do not represent the actual data for the Maanshan plant. They were, however, assumed with the view to reflecting the reliability data used in Volumes 1 and 2 of the NUREG/CR-6890 report (see [43,44]).

Table 5.1: Human error probabilities for GT1 & GT2.

Time (h)	1	2	3	4	6	7	8	10
Probability	2.07×10^{-1}	2.07×10^{-2}	3×10^{-3}	3×10^{-4}	2×10^{-4}	1×10^{-4}	1×10^{-5}	1×10^{-5}

Table 5.2: Component Reliability Data.

Component	**Transition**	**Distribution**		U_{tm}	**CCF Parameters**	
		Type	**Parameters**		**Start-up Failure**	**Running Failure**
DG-A & DG-B	1-2	Weibull	(100, 1.24)	0.009	{0.979, 0.021}	{0.972, 0.028}
	2-3	Lognormal	(6.42,2)			
	4-3	Lognormal	(5,1.2)			
GT1 &GT2	4-1	deterministic	0.5	0.0099	{0.959, 0.041}	{0.962, 0.038}
	4-2	Weibull	(200,1.5)			
	2-3	Lognormal	(5,2)			
	8-3	Lognormal	(7,1.8)			
	1-2	Weibull	(100,1.05)			
	7-4	Weibull	(0.2872,0.8194)			
	5-8	Weibull	(0.2872,0.8194)			
DG-5	1-2	Weibull	(100,1.24)			
	2-3	Lognormal	(6.42,2)			
	7-3	Lognormal	(5,1.2)			
	6-1	Weibull	(0.197,0.7467)			
	4-7	Weibull	(0.197,0.7467)			
Switchyard	4-1	Weibull	(0.197,0.7467)			
	2-1	See Figure 5.13				
Grid	2-1	See Figure 5.12				

The repair times for the six grid sources are lognormally distributed with means and corresponding standard deviations defined by $\{8.99, 11.84, 8.24, 10.25, 9.61, 9.15\}$ and $\{6.71, 4.83, 4.05, 6.61, 1.92, 5\}$ respectively. Similarly, switchyard repair times are lognormally distributed, with $\{8, 10.41\}$ and $\{5.83, 2.5\}$ respectively being the sets of means and corresponding standard deviations for the two switchyards. The effective repair distributions for the grid and switchyard nodes are modelled according to the proposal in Section 5.3.2.1, as shown in Figures 5.12 and 5.13, respectively.

All five standby generators are assumed to have a start-up failure probability of 1.756×10^{-2}. Also, the human errors associated with the failure to complete the start-up procedures for GT-5 and the switchyard are assumed equal but one-sixth of those for GT1 and GT2. Table 5.1 defines the probability of the operators not completing the start-up of the gas turbine generators within selected times. Using the procedure proposed in Section 5.3.2.3, the parameters defining transitions $7 \rightarrow 4$ and $5 \rightarrow 8$ of the gas turbine generators were obtained. The same procedure was used to obtain the parameters for transitions $6 \rightarrow 1$ and $4 \rightarrow 7$ of DG-5 and transition $4 \rightarrow 1$ of the switchyard. These and the parameters for the remaining transitions are presented in Table 5.2. The column, U_{tm}, defines the unavailability due to test/maintenance of the generators. The CCF parameters are defined by a set in which each element represents the probability of a certain number of components being involved in any failure event initiated by the component. The number of components is determined by the index of the element in the set. For instance, from the table, the probability that the start-up failure of any of the main diesel generators leads to the failure of the other generator is

Table 5.3: Common-Cause Group Definition.

CCG	Description	Attributes	
		Designation	**Value**
1	Emergency diesel generator failure to start	$\boldsymbol{\rho}^{\{1\}}$	$\{5, 6\}$
		$\boldsymbol{\theta}^{\{1\}}$	$\{0.979, 0.021\}$
		$\beta_1^{\{1\}}$	4
		$\beta_2^{\{1\}}$	3
2	Emergency diesel generator failure to run	$\boldsymbol{\rho}^{\{2\}}$	$\{5, 6\}$
		$\boldsymbol{\theta}^{\{2\}}$	$\{0.972, 0.028\}$
		$\beta_1^{\{2\}}$	2
		$\beta_2^{\{2\}}$	1
3	Gas turbine generator failure to start	$\boldsymbol{\rho}^{\{3\}}$	$\{3, 4\}$
		$\boldsymbol{\theta}^{\{3\}}$	$\{0.959, 0.041\}$
		$\beta_1^{\{3\}}$	4
		$\beta_2^{\{3\}}$	3
4	Gas turbine generator failure to run	$\boldsymbol{\rho}^{\{4\}}$	$\{3, 4\}$
		$\boldsymbol{\theta}^{\{4\}}$	$\{0.962, 0.038\}$
		$\beta_1^{\{4\}}$	2
		$\beta_2^{\{4\}}$	$\{1, 4\}$

0.021. This implies a total of two component failures, explaining why the probability value is the second element of the set (see Section 5.3.3.1 for details). Transition $4 \rightarrow 1$ of the gas turbine generators depicts their start-up duration, which as we are told in Section 5.5, takes 30 minutes, explaining why it is assigned a deterministic 0.5 hours.

5.5.3 Representing Component Interdependencies

The first and easily recognizable form of interdependency in the system is CCF, where the failure of a generator could trigger the almost instantaneous failure of another generator. This type of interdependency is modelled according to the CCF model presented in Section 5.3.3.1. DG-A and DG-B, as we know, are of the same design and model, different from the make of DG-5. Therefore, while the former are susceptible to CCF, DG-5 is immune to it. Similarly, GT1 and GT2 are susceptible to CCF, giving rise to four common-cause groups, as defined in Table 5.3. The table is developed from the CCF parameters in Table 6.1 in conjunction with the CCF model proposed in Section 5.3.3.1. CCG 1, for instance, represents the CCF due to the start-up failure of any of the main diesel generators. Since these generators are denoted as nodes 5 and 6 in the system, $\boldsymbol{\rho}^{\{1\}}$, the set of of components in the CCG is defined as $\{5, 6\}$. Now, as shown in Figure 5.8, the start-up failure of DG-A or DG-B is denoted by state 4. Also, the other generator could only be affected by this event if it is in cold standby (state 3) at the time of occurrence. This explains why $\beta_1^{\{1\}}$ and $\beta_2^{\{1\}}$ are assigned the values, 4 and 3, respectively. The parameters for CCG 2 to 4 are derived in a similar fashion.

The other form of interdependency, like the grid failure necessitating the start-up of the standby generators or the failure of GT-5 forcing the start-up of the gas turbine generators, is a little more subtle and difficult to deduce. It requires a good knowledge

of the operating principle of the system and cannot be modelled by the CCF model. For this, the cascading failure model proposed in Section 5.3.3.2 is invoked. To ensure the reproducibility of the case study, the step-by-step procedure for developing the dependency matrices is shown, by recreating the sequence of events following a LOOP.

$$\boldsymbol{D}_1 = \boldsymbol{D}_2 = \begin{pmatrix} 2 & 5 & 3 & 1 \\ 2 & 6 & 3 & 1 \\ 2 & 5 & -3 & -3 \\ 2 & 6 & -3 & -3 \end{pmatrix} \quad \boldsymbol{D}'_{10} = \begin{pmatrix} -3 & 3 & 3 & 7 \\ -3 & 4 & 3 & 7 \end{pmatrix}$$
$$\boldsymbol{D}'_5 = \boldsymbol{D}'_6 = \begin{pmatrix} -3 & 10 & 3 & 6 \\ -3 & 10 & -3 & -3 \end{pmatrix} \tag{5.17}$$

1. Let's assume the occurrence of the initiating event (LOOP), due to the failure of the grid (node 1). As already stated at the beginning of Section 5.5, the main diesel generators, A (node 5) and B (node 6), are restarted from cold standby. This is accounted for by the first 2 rows of the dependency matrix, $\boldsymbol{D}_1$. However, if the main generators are not in cold standby, maybe due to test/maintenance or failure, the shared standby generator, DG-5 (node 10), is restarted. Recalling the concept of joint dependency discussed in Section 5.3.3.2, the joint dependency between the grid and DG-5 can be deduced. Here, the main generators are the intermediate nodes, since they dictate whether or not to start the shared generator. This behaviour is jointly represented by the last two rows of $\boldsymbol{D}_1$ and the first row of $\boldsymbol{D}'_5$ in Equation 5.17. Again, if the shared generator too is unavailable (not in cold standby), the gas turbine generators, GT1 (node 3) and GT2 (node 4), are restarted (see Figure 5.10). This attribute is jointly represented by $\boldsymbol{D}'_{10}$ and the last row of $\boldsymbol{D}'_5$. If, however, the gas turbine generators are not in cold standby on arrival of their start-up signal, no action is taken. This is due to the fact that the signal signifies the unavailability of all the standby sources at the plant. $\boldsymbol{D}'_5$ and $\boldsymbol{D}'_6$ are equal because nodes 5 and 6 produce the same effect on the shared generator when unavailable for start-up. Similarly, $\boldsymbol{D}_1$ and $\boldsymbol{D}_2$ are equal, as the response of the standby systems is the same for grid and switchyard failures.

$$\boldsymbol{D}_5 = \begin{pmatrix} 2 & 6 & 3 & 1 \\ 4 & 6 & 3 & 1 \\ 2 & 6 & -3 & -3 \\ 4 & 6 & -3 & -3 \end{pmatrix} \tag{5.18}$$

2. DG-A (node 5) fails to start or starts but fails to run (see Figure 5.8). The system will first check if DG-B (node 6) is available for start-up and initiate its start up, if available. This behaviour is defined by the first two rows of $\boldsymbol{D}_5$, as shown in

Equation 5.18. The effect of the unavailability of DG-B on arrival of its start-up signal has already been defined in scenario 1 (see the last row of $\boldsymbol{D}_1$). This is adapted to account for the case when DG-A fails to start or run and DG-B is unavailable for start-up, in the last two rows of $\boldsymbol{D}_5$ (see Equation 5.18).

$$\mathbf{D}_6 = \begin{pmatrix} 2 & 5 & 3 & 1 \\ 4 & 5 & 3 & 1 \\ 2 & 5 & -3 & -3 \\ 4 & 5 & -3 & -3 \end{pmatrix} \tag{5.19}$$

3. Similarly, DG-B (node 6) fails to start or starts but fails to run. The system will first check if DG-A (node 5) is available, and initiate its start-up. The ensuing sequence of events is similar to that in scenario 2, as illustrated in Equation 5.19.

$$\boldsymbol{D}_{10} = \begin{pmatrix} 2 & 1 & 2 & 2 \\ 2 & 2 & 2 & 2 \\ 4 & 1 & 2 & 2 \\ 4 & 2 & 2 & 2 \end{pmatrix} \quad \boldsymbol{D}'_1 = \boldsymbol{D}_1 \quad \boldsymbol{D}'_2 = \boldsymbol{D}_2 \tag{5.20}$$

$$\boldsymbol{D}_3 = \begin{pmatrix} 8 & 4 & 5 & 8 \\ 8 & 4 & 7 & 4 \\ 4 & 4 & 5 & 8 \\ 4 & 4 & 7 & 4 \\ 2 & 4 & 3 & 7 \\ 2 & 4 & 2 & 2 \\ 2 & 4 & 8 & 8 \\ 2 & 4 & 5 & 5 \\ 2 & 4 & 6 & 6 \end{pmatrix} \quad \boldsymbol{D}_4 = \begin{pmatrix} 8 & 3 & 5 & 8 \\ 8 & 3 & 7 & 4 \\ 4 & 3 & 5 & 8 \\ 4 & 3 & 7 & 4 \\ 2 & 3 & 3 & 7 \\ 2 & 3 & 2 & 2 \\ 2 & 3 & 8 & 8 \\ 2 & 3 & 5 & 5 \\ 2 & 3 & 6 & 6 \end{pmatrix} \tag{5.21}$$

$$\boldsymbol{D}'_3 = \boldsymbol{D}'_4 = \begin{pmatrix} 2 & 1 & 2 & 2 \\ 5 & 1 & 2 & 2 \\ 6 & 1 & 2 & 2 \\ 8 & 1 & 2 & 2 \end{pmatrix}$$

4. DG-5 in cold standby fails to start or starts but fails to run (see Figure 5.9). In this case, any repaired emergency diesel generator is restarted first, otherwise, the gas turbine generators are restarted. The ensuing possible sequence of events are already covered by scenarios 1-3, and it is, therefore, recommended to not explicitly redefine these in $\boldsymbol{D}_{10}$, for simplicity. It is deducible that the failure of DG-5 induces the same response sequence as grid or switchyard failure. Therefore, recreating a LOOP accounts for the failure of DG-5, as expressed in Equation 5.20.

Table 5.4: Summary of the static SBO indices obtained.

LOOP Type	$p_1^{\{sbo\}}$	f_s (per yr)	$p_2^{\{sbo\}}$	% of SBO at Start-Up	Simulation Samples
Grid	0.0033	6.18×10^{-3}	0.0022	29.23	1×10^8
Switchyard	0.0035	3.65×10^{-3}	0.0153	27.97	4.5×10^7

5. GT1 (node 3) starts up successfully and enters the start-up state (see Figure 5.10). Recall, states 7 and 8 account for the time taken by the operator to initiate the start-up of the generator. However, since both GT1 and GT2 (node 4) are in the same location, they are exposed to equal delays. Hence, the transitions, $7 \rightarrow 4$ and $5 \rightarrow 8$, of GT1 and GT2 are equal. To ensure the satisfaction of this constraint, when GT1 enters state 4, GT2 too is forced to state 4 if it is in state 7 or state 8, if it is in state 5. Similarly, when GT1 enters state 8, GT2 is forced to state 8 if it is in state 5 or state 4, if it is in state 7. This behaviour is expressed by the first four rows of $\boldsymbol{D}_3$, as shown in Equation 5.21.

6. GT2 (node 4) starts up successfully and enters the start-up state. This scenario has the same effect on GT1 as scenario 5 has on GT2. Therefore, the ensuing sequence of events is accounted for by the first 4 rows of $\boldsymbol{D}_4$, as in Equation 5.21.

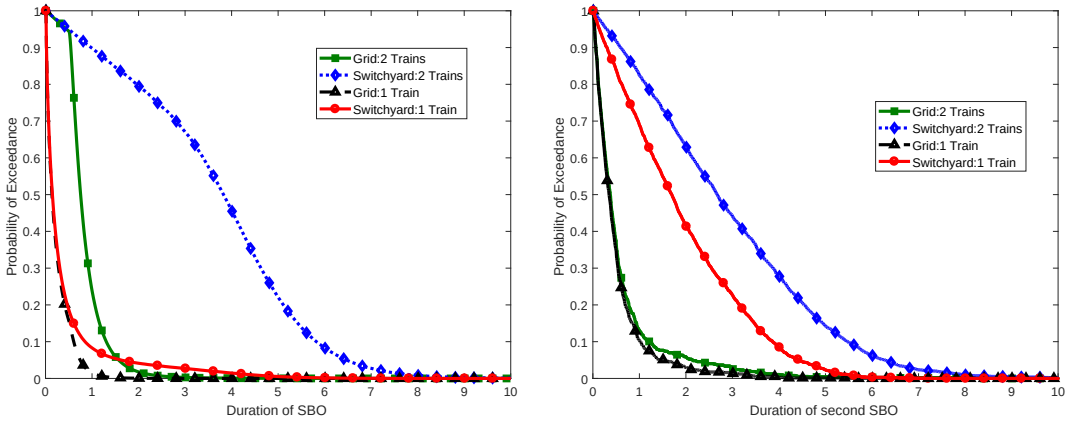
7. GT1 (node 3) fails to run. GT2 (node 4) is restarted, if it is available for start-up, otherwise the system checks whether or not the failed diesel generators have been repaired. The first case is represented by the fifth row of $\boldsymbol{D}_3$, as shown in Equation 5.21. The sequence of events involved in the second case is similar to the events following a LOOP. Therefore, a LOOP scenario is recreated, as shown in the last 4 rows of $\boldsymbol{D}_3$ and $\boldsymbol{D}'_4$. States 1, 4, and 7 have been left out of the possible GT2 states to necessitate the second case because, they mean either GT2 is already in operation (state 1), or on the verge of operation (states 4 and 7).

8. Similarly, GT2 failure to run produces the same effect on the other generators, as scenario 7. The ensuing sequence of events is, therefore, defined by $\boldsymbol{D}_4$ and $\boldsymbol{D}'_3$.

The sequence of events following the failure of the gas turbine generators to start have not been considered because, being the last standby sources to be called into operation, their start-up failure means the unavailability of the other standby sources.

5.5.4 Results and Discussions

The proposed framework is implemented in the open-source uncertainty quantification toolbox, OpenCOSSAN [109, 110] and used to quantify the SBO risk at the Maanshan nuclear power plant. For a grid and switchyard LOOP frequency of 1.86×10^{-2} and 1.04×10^{-2} per/year respectively, the case study was analysed on a 2.5GHz, E5-2670 v2 Intel ® Xeon ® CPU. A 5% coefficient of variation was imposed on the conditional probability of SBO as the simulation convergence criterion. The analysis took about



(a) First SBO (b) Second SBO

Figure 5.14: Probability of SBO duration exceedance.

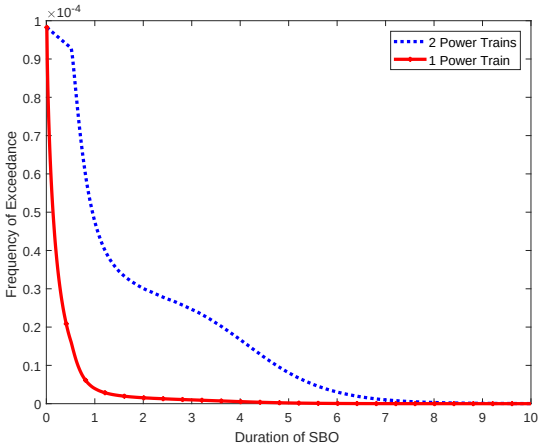
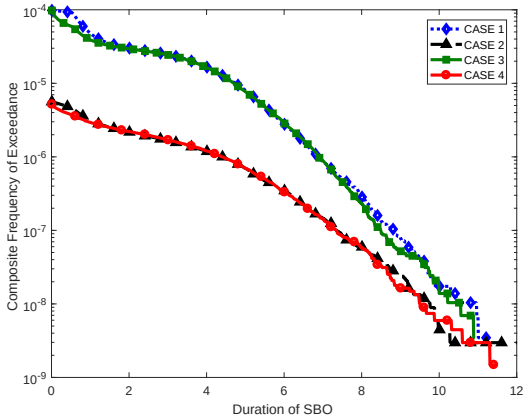


Figure 5.15: Composite frequency of first SBO exceedance.

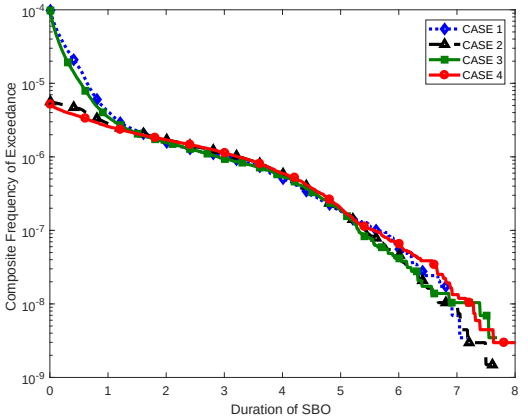
3 hours, and the results yielded are summarised in Table 5.4, Figure 5.14, and Figure 5.15. The probability of exceedance gives a measure of the likelihood of non-recovery from the SBO within a given time. The composite frequency of exceedance is the sum of the frequencies of exceedance yielded by the two LOOP categories investigated.

As shown in Table 5.4, the probability of an SBO given a LOOP is almost the same for both LOOP categories. The slight difference is due to the fact that the gas turbine generators are unusable during switchyard centred LOOP. Their effect, however, is prominent in mitigating the second SBO. The non-recovery probability from an SBO, as shown in Figure 5.14, is expressed as the non-recovery likelihood as a function of time and number of safety buses. The overall SBO risk at the plant is defined by the composite frequency of exceedance, as shown in Figure 5.15.

As a way of verifying the convergence of the simulation, the product of $p_1^{\{sbo\}}$ and the fraction of SBO at start-up, should match the probability, p_0, of the emergency power system being unavailable at time 0. Bear in mind GT-5 and the gas turbine



(a) Composite frequencies of exceedance when two power trains are required for power recovery



(b) Composite frequencies of exceedance when one power train is sufficient for power recovery

Figure 5.16: Comparison of composite frequencies of exceedance.

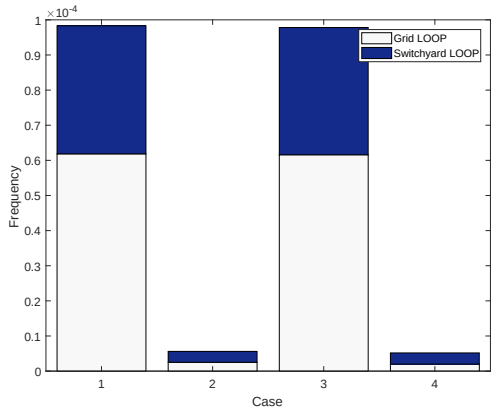


Figure 5.17: Comparing SBO frequencies.

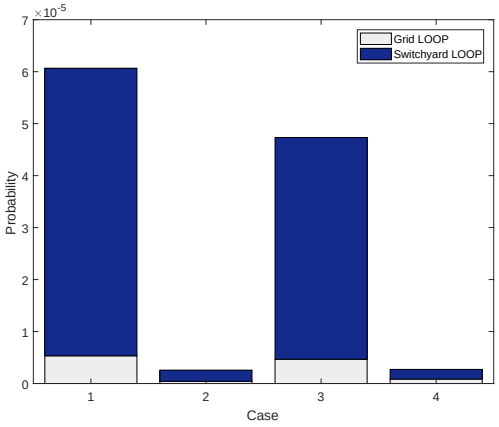


Figure 5.18: Comparing 2^{nd} SBO probs.

generators have no influence on p_0, as a result of the delays in their start-up. Therefore, the emergency power system is unavailable at start-up only if DG-A (or DG-B) is unavailable due to test/maintenance and DG-B (or DG-A) fails to start or both are not in test/maintenance but fail to start. If U_{tm} is the unavailability due to test/maintenance of DG-A and DG-B and p_s, their start-up failure probability, p_0 is obtained as,

$$\begin{aligned} p_0 &= U_{tm}\left(p_s + p_s\right) + \left(1 - U_{tm}\right) p_s^2 \\ p_0 &= 2U_{tm}p_s + \left(1 - U_{tm}\right) p_s^2 \end{aligned} \tag{5.22}$$

Substituting the required values in Equation 5.22, an error of 3.17% is realised for grid-centred LOOP and 4.7%, for switchyard-centred LOOP. Since the error in each case is not in excess of 5%, the convergence of the simulation is verified.

Ensuring an enhanced risk insight, the system was re-analysed for three additional scenarios as follows.

- Case 2: No delays in the start-up of DG-5, implying immunity to human error.
- Case 3: Gas turbine generator start-up is simultaneous with DG-A and DG-B. The generators, however, are kept in warm standby after start-up.
- Case 4: A combination of cases 2 and 3.

Case 1 represents the scenario already analysed, and the results for the four cases are summarised in Figures 5.16 to 5.18 (please note the composite frequencies in Figures 5.16 (a) and (b) are expressed on a log-scale). Absolute, instead of conditional probabilities, have been used in Figure 5.18, to ensure uniformity.

The following risk insights are inferred by the outcome of the case study;

1. As shown in Figure 5.14, station blackouts induced by switchyard failures are more difficult to recover from and, therefore, contribute more to the overall SBO risk. In this light, feasible reliability improvement programs should be designed to ensure the high reliability of the switchyard. Such a programs should be complemented by an efficient repair policy to keep the non-recovery probability low.
2. The gas turbine generators are the only difference between the recovery durations of grid and switchyard LOOP. These generators, therefore, are very instrumental to mitigating SBO risks at the plant, and their availability should be kept high.
3. Automating the start-up of DG-5 and initiating the start-up of the gas turbine generator just after LOOP guarantees an improved resilience to SBO, as endorsed by Figures 5.16 to 5.18. However, starting the gas turbine generators simultaneously with the emergency diesel generators brings with it additional costs, borne from fuel consumption and maintenance. This decision, therefore, should be preceded by a robust cost-benefit analysis. In fact, under economic constraints, it is prudent to automate the start-up of DG-5 only, as the difference between the outcomes yielded by case 2 and case 4 is only just slight.

The explicit sensitivity and importance analyses of the individual components has been ignored, since these quantities can be achieved even with the existing techniques.

5.6 Chapter Summary

Station blackout accidents, though a rare occurrence, can have devastating consequences on a nuclear power plant's ability to achieve and maintain safe shut-down. Consequently, the plant's capability to cope and recover from them is an indicator of its resilience.

In this chapter, an intuitive simulation framework to model a nuclear power plant's recovery from station blackout accidents has been proposed. The framework provides a simple means of defining the complex interdependencies that often characterise the

operation of practical engineering systems, and therefore, applicable without unrealistic assumptions. This attribute, coupled with its ability to intuitively tolerate the multi-state behaviour of the system's building block, distinguishes it from the existing approaches. Its applicability has been demonstrated by modelling the SBO recovery of a pressurised water reactor, providing an informed insight into its SBO risks. The proposed approach was able to fully model the dynamic behaviour of the power system and provide valuable insights into the SBO risk mitigation at the plant. The non-recovery probability curve obtained, for instance, can be absorbed into the existing probabilistic risk assessment models, getting rid of laborious fault trees. Since this curve also depicts the unavailability of AC power, it can be directly compared with the reliability of the plant's SBO coping mechanism, providing an easier means of determining the need for their reliability improvement. It also helps ascertain the adequacy of the plant's station blackout recovery capability, without revisiting the entire model. A key desirable feature of the proposed framework is its wide applicability, even to non-nuclear applications.

In spite of their well documented limitations relative to the proposed framework, the existing static fault tree-based models still possess attributes that give them an edge in importance, sensitivity, and uncertainty analyses. With this in mind, the proposed framework has been developed with the view to complementing their applicability, instead of serving as an explicit replacement. A clear description of how its output can be incorporated into these models, has, therefore, been included. The framework, in addition, has been implemented in the open-source uncertainty quantification toolbox developed at the Risk Institute (see [109, 110]), thereby rendering it readily available.

The multi-state model and dependency matrices proposed, create the foundation for the incorporation of additional dynamic considerations. Such considerations as the optimal number of maintenance teams on-site, emergency diesel generator failure during cold standby, optimal inspection interval, and the availability of spares, are a possibility. Efforts are underway to extend the framework to these considerations, other LOOP categories, and incorporate epistemic uncertainties.

Chapter 6

Maintenance Strategy Optimization for Complex Power Systems

6.1 Introduction

Maintenance is a necessity for most multi-component systems but its benefits are often accompanied by considerable costs. However, with the appropriate number of maintenance teams and a sufficiently tuned maintenance strategy, optimal system performance is attainable. Given system complexities and operational uncertainties, identifying the optimal maintenance strategy is a challenge. A robust computational framework, therefore, is proposed in this chapter to alleviate these difficulties. The framework is particularly suited to systems with uncertainties in the use of spares during maintenance interventions, and where these spares are characterised by delayed availability. It is provided with a series of generally applicable multi-state models that adequately define component behaviour under various maintenance strategies. System operation is reconstructed from these models using an efficient hybrid load-flow and event-driven Monte Carlo simulation. The simulation's novelty stems from its ability to intuitively implement complex strategies involving multiple contrasting maintenance regimes. This framework is used to identify the optimal maintenance strategies for a hydroelectric power plant and the IEEE-24 reliability test system. In each case, the sensitivity of the optimal solution to cost level variations is investigated via a procedure requiring a single reliability evaluation, thereby reducing the computational costs significantly. The results, which are published in [52], show the usefulness of the framework as a rational decision-support tool in the maintenance of multi-component, multi-state systems.

The proposed simulation framework can be used to identify the optimal maintenance strategy for a multi-state system prone to the range of possible operational dynamics outlined in Section 2.3.1. A detailed account of its theoretical and modelling principles is provided here, setting the tone for its wide applicability. Even though algorithms for forcing maintenance in a system characterised by limited maintenance teams have

been proposed in Chapter 4, they assume all the components of the system belong to the same maintenance group. In reality, however, the components of a system could be organised into a series of maintenance groups and their maintenance made to follow a predefined strategy. These algorithms, therefore, will be replaced by a single algorithm, proposed to both overcome their limitations and improve the efficiency of the process.

The remainder of this chapter is organised thus; the next section describes the proposed approach and outlines its advantages over the existing approaches. Section 6.2 is dedicated to defining key terms, presenting a general overview of the problem under consideration, the proposed cost model, and a description of the solution procedure. In Section 6.3, a background to the component and system models is presented. The simulation algorithm and details on how components are modelled to account for various system dynamics are also described here. Section 6.4 presents two case studies, illustrating the application of the models developed to realistic systems. Finally, a conclusion is drawn on the proposed framework in Section 6.5, with insights into its applicability.

6.1.1 Advantages of the Proposed Approach

The dependability of the optimal solution obtained from any maintenance strategy optimization scheme is determined by the accuracy of its system performance measures. This, in turn, is influenced by the suitability to the system, of the reliability modelling technique employed. These modelling techniques fall into one of two broad categories; analytical and dynamic reliability models. The former are inapplicable to certain reliability problems, especially those involving complex maintenance strategies and other dynamic considerations. When forced to suit such problems, the resulting models are often oversimplified to an extent that compromises the credibility of the outcome. In fact, most of the limitations of the existing maintenance optimization models discussed in the Section 2.3.1 are associated with analytical models.

Dynamic reliability models, on the other hand, possess sufficient flexibility to model the dynamic considerations and uncertainties that characterize the operation of realistic systems. Stochastic Petri Nets [95], Stochastic Hybrid Systems [37], and Monte Carlo simulation [55,56,97,161] are the most popular in this category. Stochastic Petri Nets, however, require the enumeration of the entire state-space of the system, which makes them infeasible for complex multi-state systems, even of moderate size. They also suffer a serious setback when the system can undergo non-Markovian transitions, in which case Tuffin et al. [134] recommend simulation. Stochastic Hybrid Systems are an emerging modelling formalism with promising prospects for dynamic reliability modelling. They are built around the Markov reward model of the system, when solicited for problems involving performance optimization or system operation cost minimization [37]. Consequently, like Stochastic Petri Nets, they are intractable for complex multi-state systems, due to their susceptibility to the state explosion conundrum. In addition, they proceed

by translating the dynamic reliability problem into a set of differential equations, which closed-form solution, in some cases, may be difficult to obtain analytically. Some researchers [15] have even had to resort to a Monte Carlo simulation approach to solving these differential equations. Given the structural complexity of most power systems and their multi-state attributes, Monte Carlo simulation, therefore, remains the most feasible approach, regardless of its higher computational intensity.

However, most Monte Carlo simulation algorithms [61, 81, 161] require state enumeration and prior knowledge of the system's structure function, path sets, or cut sets when solicited for multi-state system reliability analysis, which for complex systems, is tedious. In Chapter 3, a simple load flow-based simulation approach, applicable to any system configuration was introduced. It allows the simulation of a multi-state system without the need to define its structure function, path, or cut sets. Notably, it enables the replication of realistic system operating principles like, the shut-down and restart of components. This feature is particularly necessary if the assumption of statistical independence of components should be avoided, which, especially for systems with long maintenance durations, is desirable. Shut-down events can be a result of the unavailability of another component or loading restrictions imposed on the components themselves. When dealing with maintainable systems, it is vital to consider this form of functional interdependency between components, as the failure and preventive maintenance of most components, depend on the effective time spent in operation. Most reliability analysis approaches disregard component shut-down and restart because, it is either impossible or extremely difficult to determine the actual flow through system components. In this chapter, the approach proposed in Chapter 3 is adapted to systems with limited maintenance teams and prone to maintenance delays and other operational uncertainties. The modified approach is a credible pathway via which the system performance and reliability indices relevant to the maintenance model are derived, without making unrealistic assumptions.

Appreciating that most power systems exhibit multi-state characteristics, each system component is modelled as a semi-Markov stochastic process. The multi-state model is modified to incorporate additional stochasticity induced by the operational dynamics surrounding the system. Thus, the resultant component model is also a translation of system dynamics, from the system to the component level. This model simplifies the simulation procedure, rendering it more intuitive and generally applicable. Most importantly, the procedure supports the complex scenario where various components follow different maintenance strategies; another limitation of existing models.

6.2 Problem Formulation

Consider a multi-component system of arbitrary structure, composed of either binary-state components, multi-state components, or both. These components can undergo

corrective maintenance when in a degraded state and preventive maintenance which interval is determined by the effective time spent in operation since the last maintenance action (i.e., periods when the component is unavailable do not count). The state transition times of components may be constant or follow any probability distribution. On entering a degraded state, a component is added to the maintenance queue and its repair process follows two stages; a diagnosis stage and a stage dedicated to actual repairs. At the end of diagnosis, the maintenance team may proceed to the second stage or initiate a spares request, if spares are required. The probability of the latter happening is $p_i^{\{s\}}$, where i, a positive integer arbitrarily assigned, is the index of the component in the system. There's a delay between the initiation of a spares request and the subsequent delivery of the requested spares, which duration may vary from component to component and may again follow any probability distribution. Like corrective maintenance, preventive maintenance is prone to interruptions at a probability, $q_i^{\{s\}}$. This is realised after an average time $k_i t_{pm} \mid 0 < k_i < 1$, t_{pm} being the component's expected preventive maintenance duration, and k_i, the proportion of this time to elapse before the need for spares is realised. Whilst the crew awaits the spares, they can be assigned to another job, if there are no other idle maintenance teams.

At the system level, components are arranged into ω maintenance groups, and each group maintained by $n_{t_j} \mid j = 1, 2, ..., \omega$ maintenance teams. Under dedicated maintenance, n_{t_j} is expressed in the form, $\left(n_{1_j}, n_{2_j}\right) \mid n_{1_j} + n_{2_j} = n_{t_j}$, where n_{1_j} is the number of teams dedicated to corrective maintenance, and n_{2_j}, the number of teams dedicated to preventive maintenance. It is assumed each of these n_{t_j} teams has the expertise to maintain any of the m_j components in group j. Maintenance is outsourced, and its cost constitutes three parts; a fixed cost per unit time per maintenance team, a fixed cost per maintenance call, and a fixed cost per unit time of actual maintenance service. There are no penalty costs on the system operator for failing to meet demand but consumers only pay for the quantity of output supplied. The lost revenue accrued, with the total maintenance cost over a period, provides a measure of the performance of the system for that period. It is desired to find the maintenance strategy and the value of $n_{t_j} \forall j \in \{1, 2, ..., \omega\}$, ensuring optimum system performance. The objective is the minimization of system maintenance cost, as well as the cost incurred from unmet demand. A given strategy, therefore, is optimal if it minimizes the total cost.

There are a few attributes of the system described that pose some challenges. From a modelling point of view, the fact that the system could be multi-state and of any architecture, disqualifies most of the existing system reliability evaluation techniques (see Section 6.1.1). Similarly, the limited number of maintenance teams, the uncertainties associated with the need for spares to complete a maintenance action, and the delays in the availability of these spares, present a serious planning and scheduling dilemma. For instance, if the maintenance crew knew every preventive maintenance action would require spares, they would place a spares request in advance. Alternatively, they could

carry with them a few spares in anticipation, but this would be applicable only to non-bulky components, since there is a limit to how much could be carried. The need, therefore, for an optimal maintenance strategy cannot be overemphasised.

6.2.1 Definition of key terms

1. *Expected Output-not-supplied*: A measure of the expected amount by which the actual system output deviates from its expected level, within a given period, T_m. This quantity, in power systems, is known as the Expected Energy Not Supplied (EENS), and it accounts for the periods the system performance curve is below the load curve. If $Y(t)$ and $Y_d(t)$ respectively denote the instantaneous system output and demand, then, for a demand-driven system (i.e., $Y(t) \leqslant Y_d(t)$),

$$EENS = \int_0^{T_m} \left(Y_d(t) - Y(t)\right) dt \tag{6.1}$$

For a given system reliability problem, $Y_d(t)$ is normally known, and $Y(t)$ is computed from the system reliability analysis outcome. When obtained via Monte Carlo simulation, $Y(t)$ is defined by a collection of discrete sets of system performance levels, as a function of time. The discrete form, therefore, of Equation 6.1 should be used to compute the system EENS. Given $Y(t)$ is random, the EENS is computed as the average of the performance deficiencies of all the samples of $Y(t)$. For N Monte Carlo samples of $Y(t)$, let the i^{th} sample contain j_i performance level transitions, $y_{ik} = Y_d(t) - Y(t)$ at the k^{th} transition, and $t = t_{ik} \mid 0 \leqslant t_{ik} \leqslant T_m$;

$$\begin{aligned} EENS &= \frac{Y_0}{N} \\ Y_0 &= \sum_{i=1}^{N} \left(y_{ij_i}\left(T_m - t_{ij_i}\right) + Y_1\right) \\ Y_1 &= \sum_{k=2}^{j_i} y_{i(k-1)}\left(t_{ik} - t_{i(k-1)}\right) \end{aligned} \tag{6.2}$$

the corresponding transition time, the EENS is as defined in Equation 6.2, where y_{ij_i} and t_{ij_i} are respectively the final performance level and transition time of sample i. Alternatively, if instead of $Y(t)$ and $Y_d(t)$, only the possible system

$$\begin{aligned} \beta &= \sum_{j=1}^{\alpha} \left(\boldsymbol{P}_d, j\right) \beta_0^{\{j\}} \\ \beta_0^{\{j\}} &= \sum_{i=1}^{n} max\left(\left(\boldsymbol{C}_d, j\right) - \left(\boldsymbol{C}, i\right), 0\right) \left(\boldsymbol{P}, i\right) \\ EENS &= T_m \beta \end{aligned} \tag{6.3}$$

performance and demand levels with their corresponding occurrence probabilities are known, the EENS is computed via a different approach. Let the system exist in n distinct output levels as defined by vector $\boldsymbol{C}$, with probability of occurrence within the period, T_m, defined by vector $\boldsymbol{P}$. The expected performance deviation per unit time, β, and $EENS$ are given by Equation 6.3, where α is the number of possible demand levels, $\boldsymbol{C}_d$, the vector defining these levels, and $\boldsymbol{P}_d$, the vector specifying their corresponding probabilities of occurrence. For systems like power distribution networks with multiple load points, the effective EENS, $(EENS)_{eff}$, is given by the sum of the EENS at all the load points.

2. *Shared Maintenance*: In this strategy, the same team is assigned to perform both preventive and corrective maintenance on a component or a group of components.

3. *Dedicated Maintenance*: Unlike shared maintenance, separate teams carry out preventive and corrective maintenance on the same group of components. This implies, a failed or a component due for preventive maintenance remains unattended if its dedicated maintenance team is unavailable.

6.2.2 The Cost Model

The resultant effect of component failure, maintenance strategy, and operational dynamics on the system, is expressed in terms of the expected total loss, L, incurred. Assuming zero inflation, its components are expressed as follows.

1. Loss, L_1, due to lost output, which in turn is due to system outages, consequent of component failure. If C_0 is the cost of a unit output, L_1 is expressed as,

$$L_1 = C_0 \, (EENS)_{eff} \tag{6.4}$$

   For commercial power systems, $EENS$ is in kWh and C_0, the cost of a kWh.

2. Fixed maintenance cost, L_2, emanating from fixed wages for maintenance personnel. If each team of group j is paid r_j units of currency per unit time,

$$L_2 = T_m \sum_{j=1}^{\omega} r_j n_{t_j} \tag{6.5}$$

3. Total cost, L_3, associated with the fixed cost per maintenance action. This cost accounts for the transportation of the maintenance crew, contribution to offset

the purchasing cost of tools, or both. If m_c is the cost per maintenance action,

$$\begin{aligned} L_3 &= \sum_{i=1}^{M'} m_c \left(N_i^{\{cm\}} + N_i^{\{pm\}} \right) \\ M' &= \sum_{j=1}^{\omega} m_j \end{aligned} \tag{6.6}$$

$$\begin{aligned} L_3 = &\{m_c\}_{1\times M'} \{N_i^{\{cm\}}, N_i^{\{pm\}}\}_{M'\times 2} \{1\}_{2\times 1} \\ &\mid i = 1, 2, ..., M' \end{aligned} \tag{6.7}$$

$N_i^{\{cm\}}$ and $N_i^{\{pm\}}$ respectively the number of successful corrective and preventive maintenance actions on component i, L_3 is given by Equation 6.6, where M' is the number of maintainable components of the system. When expressed in closed form, Equation 6.6) takes the form of Equation 6.7.

$$L_4 = \sum_{i=1}^{M'} \left(C_i^{\{cm\}} t_i^{\{cm\}} + C_i^{\{pm\}} t_i^{\{pm\}} \right) \tag{6.8}$$

4. Cost, L_4, of maintaining system components; a function of the time spent by each component in maintenance and the cost per unit time of maintenance. If $C_i^{\{cm\}}$ and $C_i^{\{pm\}}$ respectively are the costs of corrective and preventive maintenance of component i per unit time, $t_i^{\{cm\}}$ and $t_i^{\{pm\}}$, its total time spent in corrective and preventive maintenance, L_4 is as given by Equation 6.8, which in closed form is,

$$\begin{aligned} L_4 &= \{1\}_{1\times M'} \boldsymbol{l} \{1\}_{2\times 1} \\ \boldsymbol{l} &= \left(\{C_i^{\{cm\}}, C_i^{\{pm\}}\}_{M'\times 2} \circ \{t_i^{\{cm\}}, t_i^{\{pm\}}\}_{M'\times 2} \right) \end{aligned} \tag{6.9}$$

The '$\circ$' operator denotes element-wise multiplication of two matrices.

$$L_5 = \sum_{i=1}^{M'} \left(C_{s_i}^{\{cm\}} s_i^{\{cm\}} + C_{s_i}^{\{pm\}} s_i^{\{pm\}} \right) \tag{6.10}$$

5. Cost, L_5, of spares used in maintaining system components. For most systems, on average, the spares used during preventive maintenance are minor and cheaper than those used in corrective maintenance. Let $s_i^{\{cm\}}$ and $s_i^{\{pm\}}$ respectively be the number of spares used in corrective and preventive maintenance of component i. If their corresponding unit costs are respectively $C_{s_i}^{\{cm\}}$ and $C_{s_i}^{\{pm\}}$, then L_5 is as expressed in Equation 6.10, which in closed form condenses to,

$$\begin{aligned} L_5 &= \{1\}_{1\times M'} \boldsymbol{l} \{1\}_{2\times 1} \\ \boldsymbol{l} &= \left(\{C_{s_i}^{\{cm\}}, C_{s_i}^{\{pm\}}\}_{M'\times 2} \circ \{s_i^{\{cm\}}, s_i^{\{pm\}}\}_{M'\times 2} \right) \end{aligned} \tag{6.11}$$

The overall system lost revenue, L, is given by,

$$L = \sum_{i=1}^{5} L_i \tag{6.12}$$

Normally, the nominal system output and the various costs are known. Determination of L, therefore, effectively reduces to the task of estimating $(EENS)_{eff}$, $\{N_i^{\{cm\}}, N_i^{\{pm\}}\}_{M' \times 2}$, $\{t_i^{\{cm\}}, t_i^{\{pm\}}\}_{M' \times 2}$, and $\{s_i^{\{cm\}}, s_i^{\{pm\}}\}_{M' \times 2}$ via reliability evaluation. These parameters are a function of the failure and maintenance events of the system components, and are therefore random. As a consequence, their mean/expected values are used in calculating the system lost revenue, L.

If the system reliability and performance indices for strategy k are represented by the function $R(\boldsymbol{n}^*, k)$, and the set of costs, by C, then, the system loss function can be expressed in the form, $L(C, R(\boldsymbol{n}^*, k))$. With $R(\boldsymbol{n}^*, k)$ known for all the possible maintenance strategies, the optimal maintenance strategy can be identified and its sensitivity to variations in cost levels investigated, without the need for multiple simulations.

6.2.3 Proposed Maintenance Regimes

Depending on the type of maintenance strategy in use, different system performance outcomes are possible, even with the same number of maintenance teams. For instance, in a series-connected system, it may seem reasonable to postpone preventive maintenance until system failure. In such a scenario, preventive and corrective maintenance actions are performed concurrently. Ideally, this should result in reduced system downtime and subsequent improvements in performance. This is normally the case if preventive maintenance actions are frequent, require large times, or if some components are not easily accessible, such that their maintenance inflicts significant throughput losses on the system. However, postponing a component's preventive maintenance may increase its likelihood of failure and bring with it additional costs. These costs are incurred from spares used, longer system down times, and higher maintenance intervention costs, as corrective maintenance durations normally are longer. In addition, more than one maintenance team may be required for efficient implementation of this strategy, since there may be multiple components requiring maintenance intervention when the system fails. On the downside, the teams are idle while the system is in operation but continue to receive salaries as the maintenance contract demands. A similar argument can be proffered for corrective maintenance of partially failed components, if in spite of the failure, system performance remains above a certain threshold. This procedure, however, may be counterproductive if component interdependencies exist in the system, such that a degraded component affects the operation of healthy ones. Therefore, even for a system this simple, it is difficult to determine whether the procedure yields the most cost effective solution, without a detailed reliability analysis. In summary, the optimality of a

given strategy depends, amongst other factors (cost levels, for instance), on the topology of the system and the non-topological functional relationships between its components.

The following regimes may be considered when deciding the promptness of preventive maintenance and major corrective maintenance of partially failed components.

1. Maintenance can be carried out at any time. The time of intervention depends only on the availability of maintenance teams.

2. Maintenance is carried out only when system output is nominal.

3. Maintenance is carried out only when a component is not in operation. This may coincide with the unavailability of the entire system or a subsystem.

When the maintenance of a component is interrupted due to delays in the availability of spares, two possible scenarios ensue.

4. The component remains shut down until spares are made available. In this case, there are no risks of incurring additional costs from failures. However, the maintenance team may be assigned to another task during the wait and there will be revenue losses as the system operates below its nominal performance level.

5. The component is put back in operation, in which case it continues to perform its normal function. This results in no loss of system output, provided it doesn't fail.

6.2.4 Solution Sequence

The regimes highlighted in Section 6.2.3 can be arranged into two groups. Regimes 1-3, define the promptness of maintenance actions and 4-5, the status of a component during maintenance interruptions. Each system component may be subjected to a combination of regimes; one from each group, giving rise to 6 possible maintenance strategies. Depending on the dynamics surrounding the operation of the system, additional strategies are applicable. For instance, on the basis of division of labour, preventive and corrective maintenance interventions could be shared or dedicated. This would lead to a total of 12 possible strategies, if considered. The corresponding component and system models are then derived for each of these strategies, in preparation for system optimization.

The optimization procedure follows a two-stage approach. In the first stage, the optimal maintenance strategy is identified by analysing each system model, with no restriction on the number of maintenance teams. For each case, the performance function, L, is determined, and the optimal strategy is identified as the one yielding the least value of L. The second stage searches for the optimal number of maintenance teams using this strategy. Here, the system is re-analysed for various values of n_{t_j}, in shared policies and various combinations of n_{1_j} and n_{2_j}, in dedicated policies. Given a component can undergo only one maintenance intervention at any instance, each n_{t_j}

is bounded by $(0, m_j)$ and $\sum_{j=1}^{\omega} n_{t_j} \leqslant M'$. In dedicated policies, both n_{1_j} and n_{2_j} are bounded by $(0, m_j)$, with the additional condition, $n_{1_j} + n_{2_j} \leqslant m_j$. Additional constraints may be imposed on the number of maintenance teams in each group, depending on the maintenance strategy and certain requirements set by the operator. For example, if two maintenance groups, i and j, have at least one component in common, $n_{t_i} + n_{t_j} \leqslant \mid \boldsymbol{\theta}_i \cup \boldsymbol{\theta}_j \mid$. The operator, under economic constraints, may also impose bounds that are less than the limits already defined on the maintenance team size.

$$\begin{gathered}(L_{max}, k_{opt}) = min\left(\{L\left(C, R\left(\infty, k\right)\right)\}^{\mho}\right) \\ k = 1, 2, ..., \mho \quad k_{opt} \leqslant \mho\end{gathered} \tag{6.13}$$

$$\begin{gathered}\left(L_{min}, \boldsymbol{n}^*_{opt}\right) = min\left(\{L\left(C, R\left(\boldsymbol{n}^*_j, k_{opt}\right)\right)\}^{\phi}\right) \\ j = 1, 2, ..., \phi \quad \boldsymbol{n}^*_{opt} \in \mathbb{N} \quad L_{min} \leqslant L_{max}\end{gathered} \tag{6.14}$$

Let $\boldsymbol{n}^* \mid \boldsymbol{n}^* = \{n_{t_1}, n_{t_2}, ..., n_{t_\omega}\}$ represent a combination of maintenance teams, and $\mathbb{N} \mid \mathbb{N} = \{\boldsymbol{n}^*_1, \boldsymbol{n}^*_2, ..., \boldsymbol{n}^*_\phi\}$, the set of all possible maintenance team combinations, with ϕ denoting their total. Deriving $\mathbb{N}$ entails obtaining, first, the set defined by the number of components in each group, such that, $\mathbb{N} = \{1, 2, ..., m_1\} \times \{1, 2, ..., m_2\} \times ... \times \{1, 2, ..., m_\omega\}$ and $\phi = \prod_{j=1}^{w} m_j$. Any combinations not satisfying the operator and maintenance-strategy-imposed constraints are removed. The optimal solution, therefore, is defined by the triplet, $\left(L_{min}, \boldsymbol{n}^*_{opt}, k_{opt}\right)$, where L_{min}, $\boldsymbol{n}^*_{opt}$, and k_{opt} are respectively the minimum system loss, the optimal maintenance team size combination, and the optimal strategy. If $R\left(\infty, k\right)$ represents the reliability/performance indices of the system under maintenance strategy k with no restrictions on the number of maintenance teams, and $\mho$, the number of strategies, Equations 6.13 and 6.14 summarize the optimization procedure. $R\left(\infty, k\right)$ is obtained by setting the number of teams in each maintenance group to the number of components in that group. For this, components belonging to multiple groups are assumed to belong to the group with the least cost per maintenance team.

Large systems often result in a large number of candidate solutions. In such cases, it is advised to exploit smart optimization techniques like, Genetic Algorithm [14, 74, 97] and Particle Swarm optimization [58]. These, however, have not been considered in this chapter, as the objective here is to provide a clear insight on the component and system modelling procedures, as well as the problem formulation from first principles.

6.3 System Reliability and Performance Analysis

In this section, a brief description of the component and system modelling procedures is presented, with details on the algorithms invoked in the reliability evaluation process. To ensure simplicity and maintain focus on the modelling procedures, a perfect maintenance

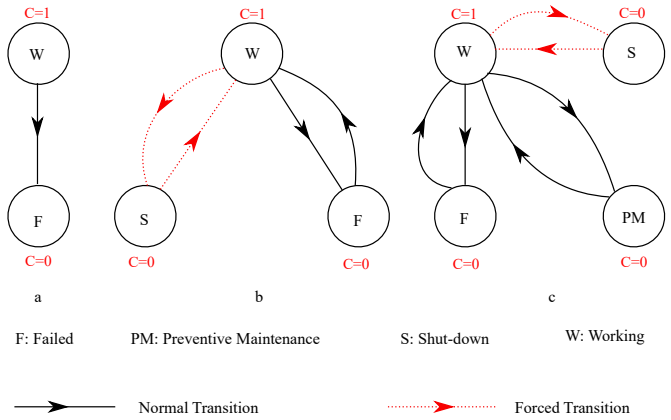


Figure 6.1: State-space of a binary-state component under various maintenance scenarios.

situation is assumed. It is, however, worthwhile noting that this is in no way limiting, as the framework can easily be extended to imperfect maintenance scenarios.

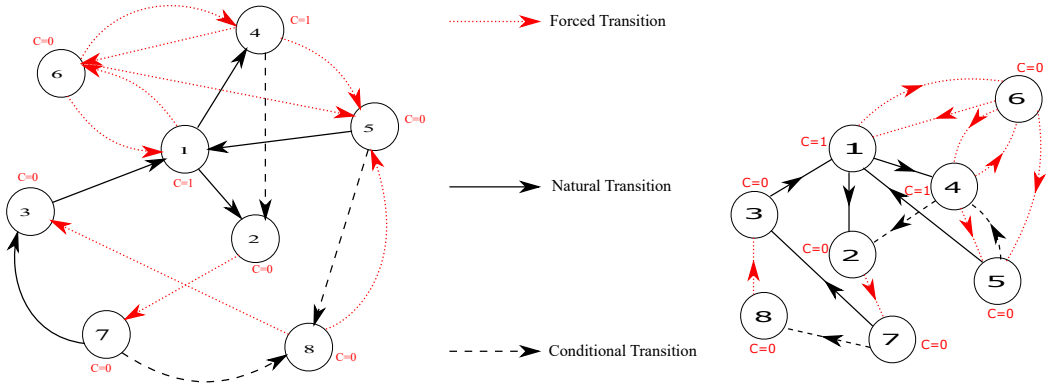
6.3.1 Component and System Representation

The multi-state model proposed in Chapter 3 is adopted to define the behaviour of each system component. This model takes cognisance of the various parameters required for the complete representation of a component's attributes. It accounts for the component's possible state transitions, their associated density functions, the performance level associated with each state, and any load restrictions imposed on the component.

The system is modelled as a graph which nodes represent the components and demand points of the system, and edges; their physical links. Defining the connectivity of the graph is a square adjacency matrix, conditioned to incorporate the efficiency of the physical links. Efficient algorithms have been proposed in Chapter 3 to deduce the system flow equations from this matrix, in a format suitable for direct computation with the interior-point algorithm [68]. Given a system state vector, the actual flow through every node can be determined by updating the flow equations matrices and applying the interior-point algorithm. The matrix representation of the system structure makes the procedure easily implementable on a digital computer. See Chapter 3 for details.

6.3.2 Maintenance Modelling of Components

Consider a hypothetical series system, composed of binary-state components (components naturally existing in only 2 states) with capacity, C, equal to 1 when working, and 0, otherwise. The effects of repairs and preventive maintenance on the state-space of each system component, without maintenance delays, uncertainties, and maintenance suspensions, are first highlighted. The resulting models are later modified and generalised for multi-state components in systems prone to maintenance delays and operational dynamics. The following maintenance scenarios are considered.



(a) Kept out of operation during spares delays (b) Restored during spares delays

Figure 6.2: Repairable binary-state component under maintenance delays.

Table 6.1: Component state assignment

State	Designation	Description
1	Working	Component operates at required capacity.
2	Failed	Component is failed and CM has yet to commence; $C = 0$.
3	CM	Component is under repairs; $C = 0$.
4	Awaiting PM	Component is due for PM but maintenance has yet to commence; $C > 0$.
5	PM	PM in progress; $C = 0$.
6	Shut-down	Component not failed but taken out of operation; $C = 0$.
7	Diagnosis	Failure is being diagnosed; $C = 0$.
8	Idle	Diagnosis is complete but the maintenance team is waiting for spares to resume maintenance. Required only if delays in the availability of spares is modelled; $C = 0$.

1. Each component of the system is non-repairable (Figure 6.1 (a)).

2. A component can be repaired when failed (Figure 6.1 (b)).

3. A component can undergo preventive and corrective maintenance (Figure 6.1 (c)).

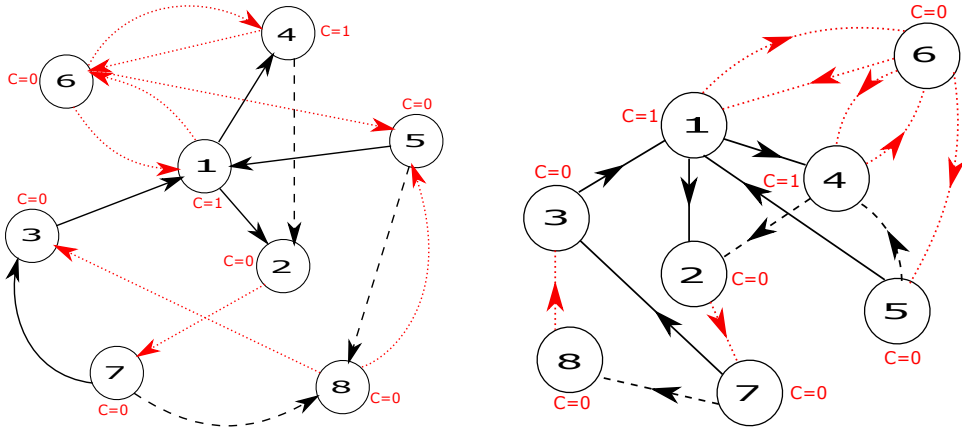
Unlike the non-repairable case, a failed component is subject to repairs in scenarios 2 and 3, which is indicated by a transition from state F to state W, in Figsure 6.1 (b) and (c). Whilst the component is in operation, other components of the system may fail. Given a series system is unavailable with the unavailability of at least one of its components, available components are unavoidably taken out of operation during repairs of failed components. A third state, S, is, therefore, introduced to account for this dependent unavailability of the operating component, as shown in Figures. 6.1 (b) and (c). The component remains in this state until all failed components are repaired, following which, it is restarted and restored. A fourth state, PM, is incorporated in Figure 6.1 (c), to represent the period the component is in preventive maintenance.

Table 6.2: Description of state transitions

Transition	Description	Transition	Description
1-2	Component Failure	7-3	Fault Diagnosis Duration
1-4	PM Interval	5-1	PM Duration
3-1	CM Duration	4-2	Failure of component whilst awaiting PM team
2-7	Forcing Diagnosis; determined by the availability of maintenance team	5-8	spares needed during PM; determined by the probability of spares being used
8-5	spares are available and PM resumes; determined by the availability of PM team	8-3	spares are available and PM resumes; determined by the availability of CM teams
7-8	Spares needed during CM; determined by the probability of spares being used	1-6	Shut-down event, like failure of system or another component
6-1,6-4	Component restart; suggests correction of event leading to shut-down	6-5	PM during shut-down; determined by availability of maintenance team and whether previous state of component was APM (state 4)
4-6	Shut-down event whilst component is due for PM	4-5	Forcing PM; determined by the availability of maintenance teams and spares
5-4	PM interruption due to spares delay		

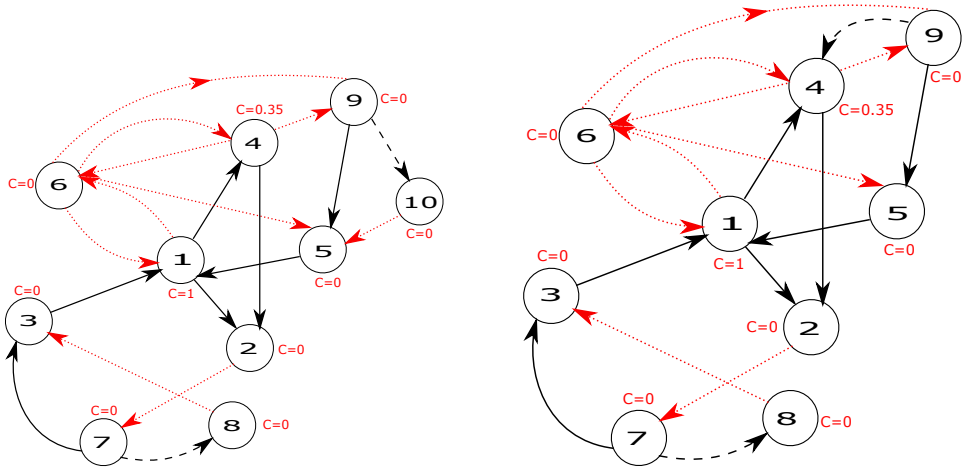
One can easily deduce that the transitions from W to F and W to PM are competing, which is due to the perfect maintenance assumption used. Since preventive maintenance and repairs make the component as good as new, any pending failures are eliminated after PM, and any scheduled PM is reset after repairs. An as good or bad as old assumption would have been implemented by replacing the transition from W to PM with a forced transition. This, however, is outside the scope of this work. It is also clear none of the three scenarios discussed considers the effects of external factors on component state transitions. For instance, there are no delays in the commencement of maintenance, and the maintenance process once initiated, suffers no obstructions or suspensions. This, however, is not the case for many practical systems.

Suppose the series system is replaced with the system described in Section 6.2, such that there are more components than there are maintenance teams. To model such case, four additional states are introduced in the state-space diagram in Figure 6.1 (c), as shown in Figure 6.2. A description of the state designations and a summary



(a) Kept out of operation during delays (b) Restored during spares delays

Figure 6.3: Binary-state component under 'maintenance only when component is unavailable'.



(a) Kept out of operation during delays (b) Restored during spares delays

Figure 6.4: Multi-state component under maintenance delays and operational uncertainties.

of the transitions depicted are presented in Tables 6.1 and 6.2 respectively. Figure 6.2 also reveals that component state transitions can be classified as either natural/normal, forced, or conditional. Natural transitions occur randomly and depend only on their associated distributions. Forced transitions occur purely as a consequence of events outside the component boundary, and their distributions are unknown. Conditional transitions on the other hand, have a known distribution but are assigned a lower priority and only occur on the fulfilment of a predefined condition or a set of conditions. In the transition matrix, $\boldsymbol{T}_i$, of the component, conditional and and forced transitions are indicated by ∞ in their relevant positions (see Chapter 3). Unlike natural transitions in which the next state of a component depends only on its current state, the next state of a component under forced transitions may also depend on its previous state. For this, a set of special procedures are defined to execute them during system simulation.

Algorithm 5 Sampling procedure for transition parameters of a multi-state component with preventive maintenance, under a limited maintenance team scenario.

Require: x and t

function SAMPLE(x)
  $\boldsymbol{J}$ ←set of possible transitions from state x
  $\boldsymbol{f}$ ←set of corresponding distributions
  k ←Number of elements in $\boldsymbol{J}$
  for n ←1 to k **do** ▷ Loop over possible transitions
    $(\boldsymbol{t}_{times}, n)$ ←Sample from $(\boldsymbol{f}, n)$
  end for
  $t_{sample} \leftarrow min(\boldsymbol{t}_{times})$ ▷ get earliest time
  $\boldsymbol{p}$ ←transitions corresponding to t_{sample}
  if $numel(\boldsymbol{p}) > 1$ **then** ▷ if multiple transitions
    $u \sim [0, 1]$ ▷ generate uniform random number
    $index \leftarrow (\boldsymbol{p}, \lceil u * numel(\boldsymbol{p}) \rceil)$
  else
    $index \leftarrow \boldsymbol{p}$
  end if
  $y_{next} \leftarrow (\boldsymbol{J}, index)$ ▷ get next state
  if y_{next} is APM **then** ▷ survives till PM is due
    $t'_{sample} \leftarrow min\left(\boldsymbol{t}_{times}, t_{sample}\right)$
    y' ←state corresponding to t'_{sample}
    $t' \leftarrow t'_{sample} - t_{sample}$
  end if
  return $(y_{next}, t_{sample}, y', t')$
end function
$t_{next} \leftarrow t_{sample} + t$

The models presented in Figure 6.2 are based on the assumption that preventive maintenance is carried out at any time or only when system performance is nominal. However, if preventive maintenance is carried out only when a component is out of service, the models are as presented in Figure 6.3. The difference between the two sets of models is the absence of the transition from state 4 to state 5 in Figure 6.3. They share the same modelling principles, as well as the designations in Tables 6.1 and 6.2.

Multi-state component modelling under maintenance delays follows a similar approach. The models in Figures. 6.2 & 6.3 can easily be generalised for multi-state components, by defining one idle state (if components are kept out of operation during spares delay), a 'Diagnosis' state (where necessary), and one corrective maintenance state for each repairable failure mode, as shown in Figure 6.4. In Figure 6.4, states 4 and 5 are a partial failure mode and its corresponding corrective maintenance state respectively. States 9 and 10 are an additional 'Diagnosis' and 'Idle' states respectively, for the partial failure mode. All the other states and transitions retain their designations and meanings, as defined in Tables 6.1 and 6.2.

6.3.3 Determining Component Transition Parameters

A system's reliability analysis by Monte Carlo simulation entails the sequential generation of the transition states and times of its components, with a view to replicating its actual operation. The next transition state, y_{next}, and time, t_{next}, of a multi-state component, are determined by which of its possible transitions from its current state, x, occurs first. Given its transition matrix, all the possible transitions from state x are sampled, and the sampled times stored in a register, $\boldsymbol{t}_{times}$. The transition corresponding to the least element of this register gives the next state of the component whilst the next transition time is given by the sum of the least element and the current simulation time, t. If multiple transitions satisfy this condition, one of them is randomly selected.

The sampling procedure described is pretty straightforward and directly applicable to most multi-state models. However, when PM is modelled as a competing transition with failures in the presence of limited maintenance teams, a slight modification to the procedure is required. For instance, if a working component is due for preventive maintenance (state 4 in Figures 6.2 and 6.3), and for some reason there is a significant delay, it may fail (transition from state 4 to 2) before the commencement of maintenance. The elapsed time depends on what the failure time would be assuming the component was not subject to preventive maintenance. Therefore, if on application of the procedure, the component is found to survive till preventive maintenance is due (i.e., its next state is APM), its next failure state, y', and the maximum period, t', it will survive before failure are also determined. This procedure is summarised by Algorithm 5.

6.3.3.1 Accounting for Non-Markovian Transitions

Algorithm 5 is only applicable to Markovian transitions (i.e., the next state of a component depends only on its current state). A second procedure, therefore, is required to implement forced and conditional transitions. The transitions to and from shut-down, with the exception of those from shut-down to CM, PM, or Diagnosis (see Figure. 6.2 to 6.4), can be implemented by the shut-down and restart procedure described in Chapter 3. The remaining conditional and forced transitions are dependent on the availability of maintenance teams or spares. For these, a procedure hinged on the assumption that the component is already assigned to an available maintenance team, is proposed.

When a component makes a transition to a new state, its next transition parameters are automatically derived, by invoking Algorithm 5. However, for the reasons already stated, this algorithm cannot derive forced maintenance transition parameters. The component's next maintenance state, y_m, from its new state, is therefore, manually determined from its transition matrix. With correct modelling according to the models proposed in Section 6.3.2, each failure mode will have at most one maintenance state (CM or Diagnosis) associated with it, and the component is added to the CM queue, if y_m exists. If on the other hand, the new state is APM, the transition parameters of the

Algorithm 6 Procedure for forcing maintenance.

Require: $p_i^{\{s\}}$, $q_i^{\{s\}}$, k_i, $s_i^{\{cm\}}$, $s_i^{\{pm\}}$, t, y_m, $\mu_i^{\{cm\}}$, $\mu_i^{\{pm\}}$

1: **function** ForceMaintenance(i, *input*)
2: $x \leftarrow y_m$ ▷ Force transition
3: $(y_{next}, t_{sample}, \sim, \sim) \leftarrow$Sample($x$)
4: **if** x is PM **then** ▷ In preventive maintenance
5: **if** $\mu_i^{\{pm\}} \leftarrow 1$ **then** ▷ If component is from suspension
6: $t_{sample} \leftarrow (\frac{1}{k_i} - 1)t_{spent}$
7: $\mu_i^{\{pm\}} \leftarrow 0$ ▷ Reset indicator
8: **else if** $u \sim [0,1] \leqslant q_i^{\{s\}}$ **then** ▷ Check whether pares needed
9: $s_i^{\{pm\}} \leftarrow s_i^{\{pm\}} + 1$ ▷ Increment PM spares used by 1
10: $t_{sample} \leftarrow k_i t_{sample}$
11: $x_{prev} \leftarrow$ previous state
12: **if** $\boldsymbol{T}_i(x, x_{prev}) \neq 0$ **then** ▷ If component should be kept out of operation
13: $y_{next} \leftarrow x_{prev}$
14: **else**
15: $y_{next} \leftarrow$'Idle' state linked to x
16: **end if**
17: $\mu_i^{\{pm\}} \leftarrow 1$ ▷ Set indicator
18: **end if**
19: **else if** x is Diagnosis **then**
20: **if** $\mu_i^{\{cm\}} \leftarrow 1$ **then** ▷ If component is from suspension
21: $x \leftarrow$ CM state connected to x
22: $\mu_i^{\{cm\}} \leftarrow 0$ ▷ Reset indicator
23: $(y_{next}, t_{sample}, \sim, \sim) \leftarrow$Sample($x$)
24: **else if** $u \sim [0,1] \leqslant p_i^{\{s\}}$ **then**
25: $s_i^{\{cm\}} \leftarrow s_i^{\{cm\}} + 1$ ▷ Increment CM spares used by 1
26: *call lines 12 to 16*
27: $\mu_i^{\{cm\}} \leftarrow 1$ ▷ Reset indicator
28: **end if**
29: **end if**
30: $t_{next} \leftarrow t_{sample} + t$
31: **return** $(y_{next}, t_{next}, s_i^{\{cm\}}, s_i^{\{pm\}}, \mu_i^{\{cm\}}, \mu_i^{\{pm\}})$
32: **end function**

component are deduced from y' and t', obtained when the algorithm was first applied when the component entered the Working state (state W). In this case, y_m is the only PM state, and the component is added to the preventive maintenance queue.

In the most general case, y_m could either be Diagnosis, CM, or PM. To force maintenance, y_m is made the current state of the component, and Algorithm 5 is applied to determine its next transition parameters. It is deducible from the component models presented in Figures 6.2 to 6.4 that a component in Diagnosis (state 7) can either undergo a normal transition to CM (state 3) or a conditional transition to Idle state

(state 8). However, the sampling algorithm always yields the normal transition. Given the conditional transition to Idle state occurs only if spares are required, a uniform random number, u, between 0 and 1, is generated and compared to the probability, $p_i^{\{s\}}$, of spares being needed to complete the maintenance. The Idle state (state 8) is made the next transition state if $u \leqslant p_i^{\{s\}}$, and the transition time yielded by the sampling algorithm is retained. In the case of repair from a partial failure mode such that the component is returned into operation during spares delay (see states 4 and 9 in Figure 6.4 (b)), the partial failure mode is made the next state, $\mu_i^{\{cm\}}$, assigned the value 1, and the component is removed from the maintenance queue. $\mu_i^{\{cm\}}$ is an indicator function that takes the value 1 when CM is suspended, and 0, otherwise.

$$\begin{aligned} t_{spent} &= k_i t_{pm} \\ t_{next} &= t + (1 - k_i)\, t_{pm} \\ &= t + \left(\frac{1}{k_i} - 1\right) t_{spent} \end{aligned} \tag{6.15}$$

Similarly, a component in PM (state 5 in Figures. 6.2 and 6.3) can either return to Working state (state 1), go to Idle state (state 8), or return to its previous state, if it should be kept in operation whilst awaiting spares. Like CM, any of the last two outcomes is determined by the probability, $q_i^{\{s\}}$, of spares being needed to complete preventive maintenance. The next transition time if spares are required is given by $t + k_i t_{pm}$, where t_{pm} is the PM duration yielded by Algorithm 5, and k_i, its proportion spent before the maintenance team realises spares are required. When PM is suspended, the component is removed from the maintenance queue, and $\mu_i^{\{pm\}}$, its indicator function for PM suspension, set to value 1. On PM resumption, the expected duration of the remainder of the maintenance exercise is $(1 - k_i)\, t_{pm}$. To avoid storing too many variables during simulation, this period is expressed in terms of t_{spent}, the time spent by the component in PM before maintenance suspension. t_{spent} is computed from the saved transition history of the component, and the next transition time is derived as in Equation 6.15. The procedure described above is summarised by Algorithm 6.

6.3.4 Maintenance Strategy Implementation

Algorithm 6 assumes the component has already been assigned an available maintenance team. However, with multiple components requiring maintenance, maintenance team assignment follows the maintenance strategy in use. Let $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$ respectively be the sets of components requiring corrective and preventive maintenance, $\boldsymbol{\Pi} = \{n_{1_j}, n_{2_j}\}_{\omega \times 2} \mid j = 1, 2, ..., \omega$, the matrix defining the number of corrective and preventive maintenance

teams in each maintenance group, and $\boldsymbol{\nu} = \{\nu_j\}_{\omega \times 1}$, an indicator vector which elements

$$\nu_j = \begin{cases} 1, & \text{If maintenance group j is shared} \\ 0, & \text{Otherwise} \end{cases} \tag{6.16}$$

are matched to the rows of $\boldsymbol{\Pi}$. Each indicator element specifies whether its corresponding maintenance group practices shared maintenance, as defined by Equation 6.16.

Given the assumption of a component being as good as new after PM or CM and the additional constraint that the former is carried out only on the perfect component, the condition $\boldsymbol{h}_1 \cap \boldsymbol{h}_2 = \varnothing$ is imposed. Therefore, prior to maintenance team assignment, all the elements of $\boldsymbol{h}_1 \cap \boldsymbol{h}_2$ are removed from $\boldsymbol{h}_2$ (i.e., $\boldsymbol{h}_2 = \boldsymbol{h}_2 - (\boldsymbol{h}_1 \cap \boldsymbol{h}_2)$ or simply $\boldsymbol{h}_2 = \boldsymbol{h}_2 - \boldsymbol{h}_1$). Depending on the maintenance strategy, additional components may be removed from $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$. For instance, if $\boldsymbol{\delta}$ is the set of components in shut-down state, $\boldsymbol{\vartheta}_1$, the set of components repairable only while in shut-down, and $\boldsymbol{\vartheta}_2$, the set of components which PM is initiated only when in shut-down, then, $\boldsymbol{h}_1 = (\boldsymbol{h}_1 - \boldsymbol{\vartheta}_1) \cup (\boldsymbol{\delta} \cap \boldsymbol{\vartheta}_1)$ and $\boldsymbol{h}_2 = (\boldsymbol{h}_2 - \boldsymbol{\vartheta}_2) \cup (\boldsymbol{\delta} \cap \boldsymbol{\vartheta}_2)$. Similarly, let $\boldsymbol{\gamma}_1$ be the set of components repairable only while system performance is nominal, and $\boldsymbol{\gamma}_2$, the set for which PM is initiated only at nominal system performance. If system performance is below nominal at maintenance team assignment, $\boldsymbol{h}_1 = \boldsymbol{h}_1 - \boldsymbol{\gamma}_1$ and $\boldsymbol{h}_2 = \boldsymbol{h}_2 - \boldsymbol{\gamma}_2$. It is, however, worthwhile to note that $\boldsymbol{\vartheta}_1$ applies to partially failed components only.

With $\boldsymbol{h}_{1f}$ and $\boldsymbol{h}_{2f}$ denoting the final contents of $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$ respectively, the first maintenance group is considered. Its assigned components in the maintenance queue are ranked according to the predefined priority rule and the top ranked component is assigned to the first available team in the group. As a consequence, the number of available maintenance teams and the number of ranked components reduce by 1 each. The procedure is repeated until all the ranked components have been assigned or until there are no available maintenance teams in the group. At this stage, $\boldsymbol{h}_{1f}$ and $\boldsymbol{h}_{2f}$ are updated accordingly, and the next maintenance group considered if $\boldsymbol{h}_{1f} \cup \boldsymbol{h}_{2f} \neq \varnothing$. This recursive procedure continues until all the maintenance groups have been covered.

Let $\boldsymbol{\theta}_j^{\{cm\}}$ be the set of components assigned to maintenance group j for CM and $\boldsymbol{\theta}_j^{\{pm\}}$, the set assigned for PM. If $\lambda_j^{\{cm\}}$ and $\lambda_j^{\{pm\}}$ are respectively the numbers of unavailable teams from group j for CM and PM, Algorithm 7 summarises the maintenance strategy implementation procedure. Line 10 accounts for when components maintained only while system performance is nominal are removed from the queue, following the deviation from nominal performance. This normally is a consequence of preventive or corrective maintenance of a partially failed component of a higher priority in the queue.

Algorithm 7 Procedure for maintenance strategy implementation.

Require: $(\boldsymbol{h}_{1f} \cup \boldsymbol{h}_{2f}) \neq \varnothing$, $\boldsymbol{h}_1$, $\boldsymbol{h}_2$

1: **for** $j \leftarrow 1$ to ω **do** ▷ Loop over maintenance groups
2: **if** $\nu_j > 0$ **then** ▷ Check whether maintenance is shared
3: $Teams \leftarrow \boldsymbol{\Pi}(j,1) + \boldsymbol{\Pi}(j,2) - \left(\lambda_j^{\{cm\}} + \lambda_j^{\{pm\}}\right)$
4: $X_{comp} \leftarrow \left(\boldsymbol{h}_{1f} \cap \boldsymbol{\theta}_j^{\{cm\}}\right) \cup \left(\boldsymbol{h}_{2f} \cap \boldsymbol{\theta}_j^{\{pm\}}\right)$
5: **while** $Teams > 0$ **and** $X_{comp} \neq \varnothing$ **do**
6: $i \leftarrow$top ranked component
7: ForceMaintenance(i, *input*)
8: $Teams \leftarrow Teams - 1$
9: $X_{comp} \leftarrow X_{comp} - i$ ▷ Remove component from maintenance queue
10: *adjust* X_{comp} *if necessary*
11: **end while**
12: **else**
13: $\boldsymbol{H} \leftarrow \{\boldsymbol{h}_{1f}, \boldsymbol{h}_{2f}\}$
14: $\boldsymbol{G} \leftarrow \{\boldsymbol{\theta}_j^{\{cm\}}, \boldsymbol{\theta}_j^{\{pm\}}\}$
15: $\boldsymbol{I} \leftarrow \{\lambda_j^{\{cm\}}, \lambda_j^{\{pm\}}\}$
16: **for** $k \leftarrow 1$ to 2 **do**
17: $X_{comp} \leftarrow (\boldsymbol{H}, k) \cap (\boldsymbol{G}, k)$
18: $Teams \leftarrow \boldsymbol{\Pi}(j,k) - (\boldsymbol{I}, k)$
19: *call lines 5 to 11*
20: **end for**
21: **end if**
22: *Remove assigned components from* $\boldsymbol{h}_{1f}$ *and* $\boldsymbol{h}_{2f}$
23: **if** $(\boldsymbol{h}_{1f} \cup \boldsymbol{h}_{2f}) \leftarrow \varnothing$ **then**
24: **break**
25: **end if**
26: **end for**
27: *Remove assigned components from* $\boldsymbol{h}_1$ *and* $\boldsymbol{h}_2$

6.3.5 The Simulation Procedure

A discrete-event simulation model is proposed to replicate the behaviour of the system. Starting with components in their initial states, the initial performance level of the system is computed and recorded. Following which, the next transition parameters of each component are sampled, and the simulation progresses to the earliest transition time. At this time, the current state of the appropriate component making the transition is updated, its new state recorded as a function of time, its next transition parameters sampled, and the next simulation time determined. This procedure is repeated for subsequent transitions until the mission time is exceeded. For every transition resulting in a change in the flow properties of a component, the output of the system is computed and recorded as a function of time. The relevant reliability and performance indices are then determined from the saved component transition and system output histories.

Let $\boldsymbol{\tau}$ be the vector of next transition times of nodes (components and output points) and $\boldsymbol{\tau}_{spare}$, the vector holding the availability times of component spares. If M is the total number of system nodes, the simulation procedure is summarised as follows.

1. Initialise the system in preparation for simulation. This involves the following:
   (a) initialization of registers to save the current flow properties of nodes, transition history of components, and the performance histories of output nodes.
   (b) setting the required number of simulations, N, and mission time, T_m.
2. Set $t = 0$, $s_i^{\{cm\}} = s_i^{\{pm\}} = \mu_i^{\{cm\}} = \mu_i^{\{pm\}} = 0 \forall i \in \{1, 2, ..., M\}$, $\boldsymbol{h}_1 = \boldsymbol{h}_2 = \varnothing$, $\boldsymbol{\tau} = \boldsymbol{\tau}_{spare} = \{\infty\}^M$.
3. Save the initial states of components.
4. Compute and save the initial performance level of all the output nodes.
5. Sample the next transition parameters of nodes, update $\boldsymbol{\tau}$, and set $t = min\,(\boldsymbol{\tau})$.
6. Check for nodes with next transition time equal to t and for each node, i,
   (a) effect the required transition.
   (b) with the exception of the case when the new state is APM, Idle, or Partial Failure given its previous state is Diagnosis, sample its next transition parameters, y_{next} and t_{next}, and determine y_m, where applicable. Update $\boldsymbol{h}_1$ or $\boldsymbol{h}_2$ if y_m exists, set $\mu_i^{\{cm\}}$ and $\mu_i^{\{pm\}}$ to 0, and go to Step (g).
   (c) if the new state is APM, $y_{next} = y'$, $t_{next} = t' + t$, y_m is set to the PM state, and $\boldsymbol{h}_2$ updated. However, $\boldsymbol{h}_2$ is not updated if the node is returning from PM, as the transition depicts a maintenance suspension. In this case, $t_{next} = t_{rem} + t$, where t_{rem} is the remaining life of the component prior to its maintenance being forced. Go to Step (f).
   (d) if the new state is Partial Failure and previous state, Diagnosis, $t_{next} = t_{rem} + t$, the expected failure state before the transition to Diagnosis is made y_{next}, and y_m is set to Diagnosis. Go to Step (f).
   (e) if the new state is Idle, $t_{next} = \infty$. y_m is set to PM if the node is from PM, and CM, if it is from Diagnosis. Go to Step (f).
   (f) steps (d) and (e) involve maintenance suspensions. For these and the case involving PM suspension in step (c), the time, t_{spare}, the spares will be delayed by, is sampled from the appropriate distribution. Update $\boldsymbol{\tau}_{spare}$, such that $(\boldsymbol{\tau}_{spare}, i) = t_{spare} + t$.
   (g) update the node's state history, the flow vectors, and $\boldsymbol{\tau} \mid (\boldsymbol{\tau}, i) = t_{next}$.

7. Identify the nodes which spares have been made available. For each node, i, update $\boldsymbol{\tau}_{spare} \mid (\boldsymbol{\tau}_{spare}, i) = \infty$, $\boldsymbol{h}_1$, if y_m is CM or Diagnosis, and $\boldsymbol{h}_2$, otherwise.

8. Compute $\boldsymbol{h}_{1f}$ and $\boldsymbol{h}_{2f}$ and call Algorithm 7.

9. If the current and previous flow property vectors differ:

   (a) restart nodes in shut-down, compute the flows through the nodes of the system, and shut down nodes, as proposed in Chapter 3.

   (b) for each output node, update its performance history if its current and previous performances differ.

10. Save the current node flow property vectors.

11. Compute $\boldsymbol{h}_{1f} = h1 \cap \boldsymbol{\delta} \cap \boldsymbol{\vartheta}_1$ and $\boldsymbol{h}_{2f} = h1 \cap \boldsymbol{\delta} \cap \boldsymbol{\vartheta}_2$ and call Algorithm 7 again. This step accounts for those components maintainable only while in shut-down.

12. Set the next simulation time, $t = min\left(min\left(\boldsymbol{\tau}\right), min\left(\boldsymbol{\tau}_{spare}\right)\right)$.

13. Repeat Steps 6 to 12 until $t > T_m$, updating $\boldsymbol{\tau}$, the flow property vectors, node state histories, and output performance histories at every transition.

14. Repeat Steps 2 to 13, N times, saving the final node histories at every trial.

15. Determine the system performance indices.

The desired performance indices are, $(EENS)_{eff}$, $\{N_i^{\{cm\}}, N_i^{\{pm\}}\}_{M' \times 2}$, $\{t_i^{\{cm\}}, t_i^{\{pm\}}\}_{M' \times 2}$, and $\{s_i^{\{cm\}}, s_i^{\{pm\}}\}_{M' \times 2}$. The last set of indices is yielded directly by the simulation algorithm, $(EENS)_{eff}$ is computed from the performance histories of output nodes, and the remainder, from the state transition histories of components. $t_i^{\{pm\}}$ is given by the average time spent by component i in preventive maintenance (e.g., state 5 in Figs. 6.2 and 6.3), $t_i^{\{cm\}}$, the average time spent in Diagnosis and corrective maintenance (e.g., states 7 and 3 in Figures 6.2 and 6.3, states 3, 5, 7 and 9 in Figure 6.4), $N_i^{\{cm\}}$, the average number of transitions from all CM states to Working state (e.g., transition 3-1 in Figsures 6.2 and 6.3, transitions 3-1 and 5-1 in Figure 6.4) and $N_i^{\{pm\}}$, the average number of transitions from PM state to Working state (e.g., transition 5-1 in Figures 6.2 and 6.3). These indices are substituted in the equations proposed in Section 6.2.2, to compute the system loss.

The simulation procedure, with its associated algorithms, accounts for most of the forced and conditional transitions. As a result, some of these transitions could be omitted from the component models without compromising the simulation outcome. For instance, the shut-down state and its related transitions could be omitted altogether. This, however, does not mean shut-down and restart are not accounted for during simulation. Of the remaining forced and conditional transitions, only those to and from

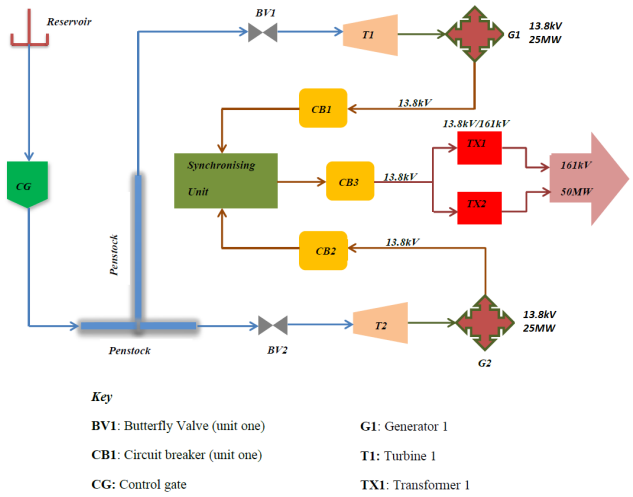


Figure 6.5: Schematic of a 2-unit hydroelectric power plant.

Table 6.3: Component and system data for the hydroelectric power plant.

Component	Valves	Turbines	Gens.	Breakers	Synch.	Xfmr.
Failure time distribution	Wb(1000,1.5)	Wb(4125,2.1)	Wb(2000,2)	Exp(3750)	Exp(3250)	Exp(2500)
Repair time distribution	Exp(40)	LogN(106,5)	Exp(150)	Exp(36)	Exp(96)	Exp(80)
PM interval	U(500,625)	U(1125,1250)	U(1125,1250)	U(2125,2175)	U(2125,2175)	U(2125,2175)
PM duration	Exp(8)	Exp(21.2)	Exp(30)	Exp(7.2)	Exp(19.2)	Exp(16)
Diagnosis duration	Exp(5)	Exp(14)	Gu(20,3.24)	G(5,2)	Exp(16)	LogN(16,2)
Spares cost(CM)	1624	2100	1944	1006	2245	2700
Spares cost(PM)	1055.6	1365	1263.6	653.9	1459.25	1755
PM cost/hr	162.5	243.75	203.13	101.56	243.75	264.06
CM cost/hr	250	375	312.5	156.25	375	406.25
Spares delay	Exp (24)					
Probability of Component Replacement During Maintenance						
CM (p_i)	0.5	0.55	0.8	0.9	0.7	0.6
PM (q_i)	0.8	0.9	0.96	0.42	0.4	0.45
Mean Fraction of PM Duration Before Component Replacement Becomes Eminent						
Fraction (k_i)	0.25	0.25	0.25	0.25	0.25	0.25

Diagnosis state, from PM to Idle state, and from PM to APM state (if applicable) are required, the rest could be omitted. Applying this new information to the component models presented in Figures 6.2 to 6.4, for instance, would result in much simpler models.

6.4 Case-Studies

The proposed framework is implemented in the MATLAB-based toolbox, OpenCosan [109, 110], and used to identify the optimal maintenance strategies for two power systems.

6.4.1 Case-Study 1: A 50MW Hydroelectric Power Plant

In this case-study, a two-unit hydroelectric power plant is analysed. It is a slightly modified model of the Bumbuna hydroelectric power plant, a 50MW plant in Sierra Leone. Its two units are similar, and each, rated 25MW consists a butterfly valve, turbine, generator, and circuit breaker. Their generated power is synchronized in the

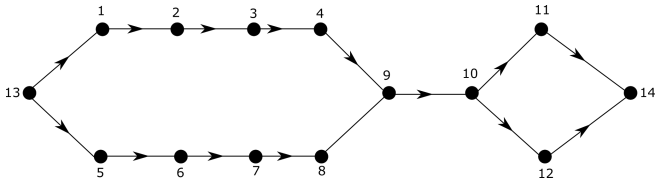
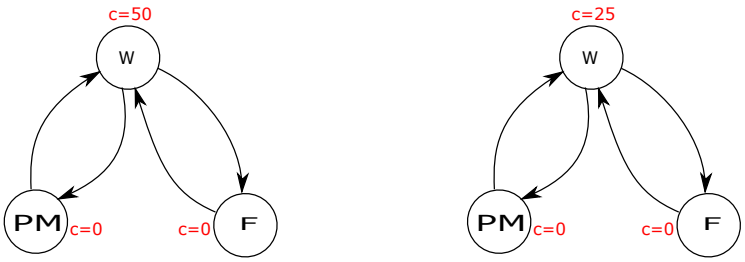


Figure 6.6: Plant's network model.



(a) Nodes 9 and 10 (b) Others but 13 and 14

Figure 6.7: State-space diagrams of components.

synchronizing unit and fed to the step-up transformers for onward transmission. These transformers are also rated 25MW, and when one is unavailable, the plant is reconfigured such that only one unit operates. The plant's schematic representation is shown in Figure 6.5 and its reliability data, in Table 6.3. All failure and repair times are in hours, and costs, in British Pounds (£). The unit cost of electricity is £ 0.5, the fixed wage per maintenance team is £ 7 per hour, and a negligible cost per maintenance call. It is worthwhile noting that the data presented in Table 6.3 are assumed, and therefore for illustrative purposes only. Ideally, such data should be based on actual field data extracted from component failure and maintenance histories.

6.4.1.1 Modelling the Plant and its Components

The following assumptions are considered.

1. All components operate at only two distinct performance levels.
2. Components are ranked in their order of arrival in the maintenance queue.
3. There is only one maintenance group.
4. The load is fixed at 50MW, and the reservoir can always meet this demand.
5. The failure rates of the control gate and penstock are negligible.

Figure 6.6 shows the network model of the plant. The components of unit 1, that is, valve-1, turbine-1, generator-1, and breaker-1 are respectively denoted by nodes 1 to 4 and their counterpart in unit 2, by nodes 5 to 8. Nodes 9 to 14 respectively represent the synchronizer, breaker-3, transformer-1, transformer-2, dam, and the external load.

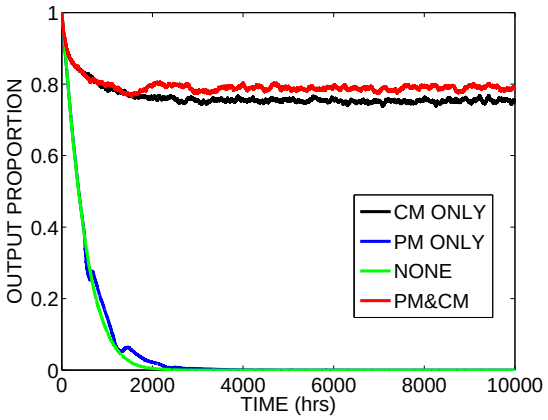


Figure 6.8: Plant output performance.

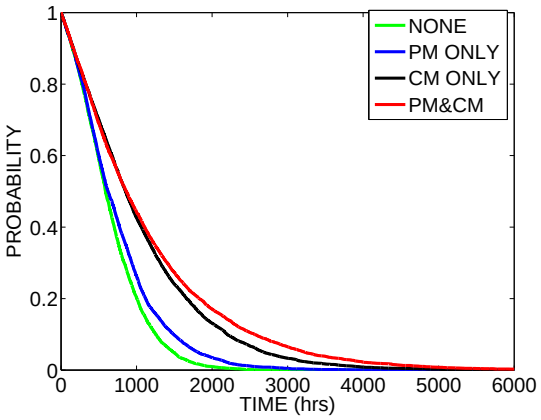


Figure 6.9: Plant reliability.

Assuming perfect links between components, the parameters of the network are obtained as proposed in Chapter 3. For this system, the number of nodes, M, is 14 while the number of maintainable components, M', is 12. The state-space diagrams of the components, without maintenance delays are shown in Figure 3.8. Under the range of possible maintenance regimes proposed in Section 6.2.3, these state-space diagrams can be transformed into those in Figure. 6.2 and 6.3. Since the demand and source (dam) capacity are fixed at 50MW, nodes 13 and 14 have a single state of capacity 50 units.

The reconfiguration procedure used in the simulation shuts down nodes when their load flow drops below a threshold. To enable plant reconfiguration when only one transformer is available, a minimum load restriction is imposed on the turbines. The choice of the turbines, however, is arbitrary, as any of the unit nodes would do, due to their being connected in series. With only node 11 or 12 available, the load flow from node 13 drops to 25MW, which is divided equally between the two units if they both are in operation. The threshold flow for each turbine, therefore, is set to a value slightly greater than 12.5 units (say 12.52), and 0, for all the other nodes.

6.4.1.2 The Effects of Maintenance on System Performance and Reliability

The plant is analysed separately under the assumptions that its components are non-repairable, subject to preventive maintenance only, corrective maintenance only, and both maintenance types. With the exception of the non-repairable case, there is no restriction on the number of parallel maintenance actions that can take place. The maintenance team size in each case, therefore, is expressed as (0 0), (0 12), (12 0), and (12 0), respectively. Dedicated maintenance is used in the second and third cases to ensure only the intended maintenance type is carried out (e.g., no CM during a PM only policy). This stage is aimed at investigating the relative effects of the various maintenance strategies on the plant's reliability, performance, and loss function. It identifies the candidates for the optimal maintenance strategy and determines whether

Table 6.4: Plant expected output and loss.

Strategy		Output (GWh)	L ($£10^6$)
None		23.66	238.17
PM only		26.06	237.82
CM only		382.21	60.98
PM+CM	[1,4]	370.99	66.38
	[1,5]	384.21	59.91
	[2,4]	369.18	67.51
	[2,5]	383.57	61.42
	[3,4]	396.29	53.63
	[3,5]	388.22	58.07

or not to proceed with the search for the optimal maintenance team size. This prevents searching in unlikely regions or strategies, thereby reducing the computational cost.

Figures 6.8 and 6.9 respectively show the reliability and instantaneous performance of the plant as a fraction of its nominal output, for a mission time of 1×10^4 hours and 5×10^3 Monte Carlo samples. Plant reliability is defined with respect to complete outages, however, excluding those due to preventive maintenance (scheduled outages). The objective is to study the survivability of the plant, which scheduled outages would underestimate. For instance, more frequent outages may be experienced under a maintenance strategy incorporating both preventive and corrective maintenance than one with the latter only. In practice, scheduled outages do not count toward a systems's survivability, since they are out of choice rather than failure. Hence, the need for their disregard in its survivability analysis. In summary, plant reliability at time, t, is the non-occurrence probability of complete-outage-inducing failures in the interval $[0, t]$.

The reliabilities and instantaneous performances defined by Figures 6.8 and 6.9 depict the upper bounds for the various maintenance strategies. As expected, both maintenance types, indeed, improve the reliability and performance of the plant. The impact of PM, however, is only slight, given that 50% of the components exhibit an exponential failure characteristic. For such components, PM only reduces their availability without an improvement in reliability [161]. Preventive maintenance, therefore, is most effective in systems with ageing components. Table 6.4 presents the upper bound of the expected plant output and the corresponding loss for each maintenance strategy. The notation [a,b] denotes a strategy made up of a combination of regimes a and b, as described in Section 6.2.3. A review of the trend portrayed in Figures 6.8, 6.9, and Table 6.4, suggests a maintenance strategy incorporating both PM and CM is desirable. The losses in Table 6.4 are yielded by the maximum number of maintenance teams, the optimal loss in each case, therefore, is provided by fewer maintenance teams. These teams are determined by the procedure proposed in Section 6.2.4.

Table 6.5: Optimal plant loss as a function of maintenance strategy

Strategy	L ($£10^6$)	Number of teams
$[1,4]$	65.66	2
$[1,5]$	59.24	2
$[2,4]$	66.88	3
$[2,5]$	59.65	3
$[3,4]$	52.89	5
$[3,5]$	57.32	4
CM only	60.14	4

Table 6.6: Optimal maintenance strategy sensitivity to costs.

	Cost Element			
	EC	FMC	CPHM	CS
Strategy	$(0\ 0)$, $k_f = 0$ $[3,4]$ $k_f > 0$	$[3,4]$ $\forall k_f$	$[3,4]$, $0 \leqslant k_f < 70.9$ $[3,5]$, $k_f \geqslant 70.9$	$[3,4]$ $\forall k_f$

6.4.1.3 Optimal Maintenance Strategy Identification

It is clear the non-repairable and preventive-maintenance-only strategies are very inefficient. The plant, therefore, is analysed for the other strategies, using the same mission time and number of samples as in the preceding section. The optimal solution for each strategy is identified and recorded as shown in Table 6.5. From these, the best maintenance strategy and the optimal number of maintenance teams are deduced as [3,4] and 5, respectively. To explore the existence of a more optimal solution for this strategy, the plant is re-analysed under dedicated maintenance. It is observed that for the same number of teams, shared maintenance strategies produce a better plant performance.

The optimal strategy being [3,4] is in agreement with the preliminary results presented in Table 6.4. Therefore, the optimal solution would have been obtained using this strategy alone. However, the other strategies were considered to establish a relationship between the optimal maintenance team size and maintenance strategy.

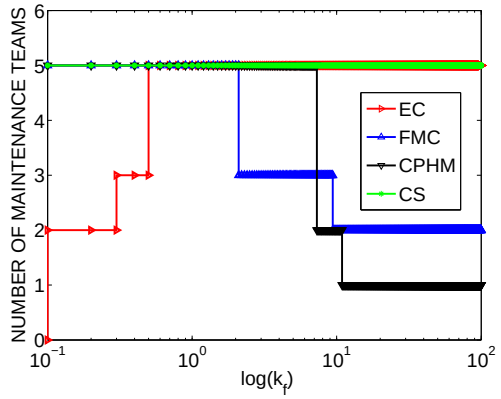


Figure 6.10: Optimal maintenance team size sensitivity to costs.

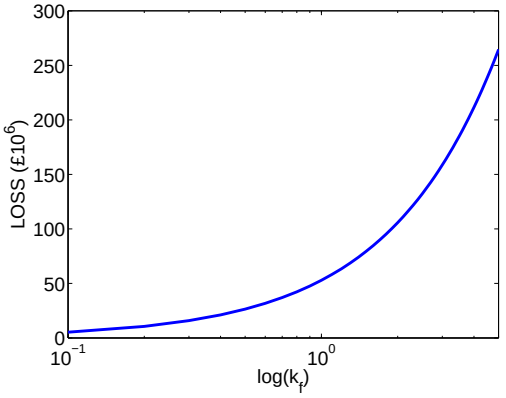


Figure 6.11: Optimal system loss sensitivity to cost-level variation.

6.4.1.4 Sensitivity to Cost Levels

The robustness of the optimal maintenance strategy to variations in cost of electricity (EC), fixed cost per maintenance team (FMC), fixed cost per hour of maintenance (CPHM), and cost of spares (CS) is investigated. Figure 6.10 shows how the number of maintenance teams required for optimal performance varies with $k_f \mid 0 \leqslant k_f \leqslant 100$, where k_f is the ratio of new cost to the original cost provided in Table 6.3. It is evident from the figure that the optimal maintenance team size is insensitive to the cost of spares but exhibits a fair degree of sensitivity to the other costs. In contrast, the optimal maintenance strategy is insensitive to all four cost elements up to $k_f = 70.9$ (for CPHM), beyond which [3,5] becomes the optimal strategy, as shown in Table 6.6.

In practice, when inflation occurs, it affects all the cost elements concurrently. The sensitivity of the optimal solution in such a scenario is investigated. It is observed that with $k_f = 0$, the maintenance strategies are all equivalent, since all the services are in effect provided free-of-charge. Beyond this value, the optimal maintenance strategy and the number of teams remain constant at [3,4] and 5, respectively, for the entire range of k_f. The optimal loss, however, increases according to Figure 6.11. This strange behaviour is explained by the dominance of the cost of electricity in the loss equation (see Section 6.2.2). When all the four costs change by the same factor, the resultant effect is dominated by the electricity cost, for $k_f > 0.4$, and the other costs, otherwise. A comparison of the trends portrayed in Figures 6.10 and 6.12 supports this theory. Figure 6.12 is obtained by holding fixed, the cost of electricity and varying the maintenance costs. Expectedly, it shows a decrease in the optimal maintenance team size, with rising maintenance costs. Indeed, with high maintenance costs, the only logical decision is downsizing the maintenance team, to ensure sustainability.

6.4.1.5 Computational costs

The simulations were run on a 48 core, 1895.257MHz AMD Opteron(tm) 6168 processor, using 19 cores running in parallel. Less than one minute was required for the non-repairable system and an average of 8.95 minutes per candidate solution for the system under preventive and corrective maintenance.

6.4.1.6 Discussions

Analytical approaches do not make a feasible option for the analysis of complex systems with realistic attributes. Simulation algorithms, on the other hand, are disadvantaged by their large computational costs, made worse when employed in optimization procedures. This, often, is attributed to the large number of samples required for a dependable estimate of the system performance indices. Therefore, the trade-off between accuracy and moderate computational burden is worth adequate attention. Another limiting constraint of great importance is the mission time, which should be selected such that,

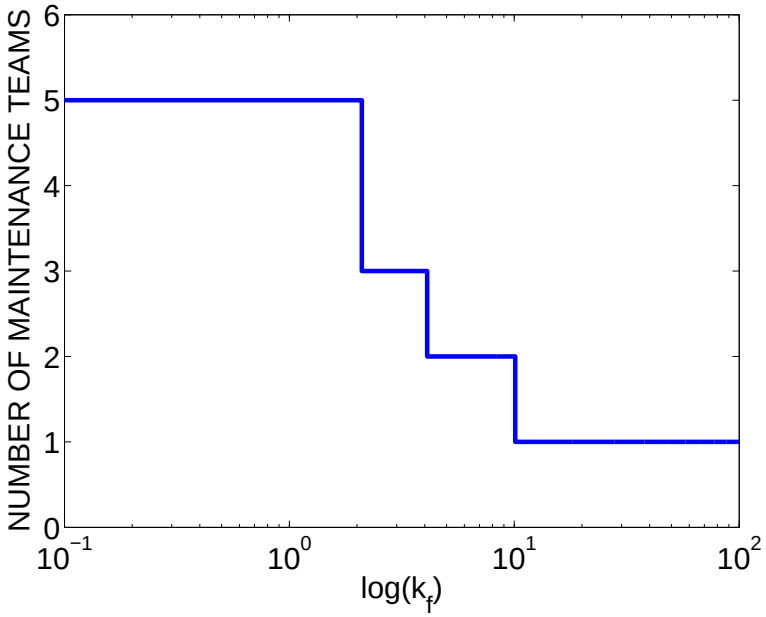


Figure 6.12: Sensitivity of optimal solution to concurrent variation in FMC and CPHM.

the performance indices obtained reflect the true long-term indices of the system. This requires that the mission time be sufficiently greater than the time the system takes to attain steady-state. In the case study presented, 5×10^3 samples are just enough to provide an acceptable degree of accuracy and a manageable computational burden. Also, from Figure 6.8, the plant's steady-state attainment time is about a fifth of its mission time. These attributes endorse the dependability of the optimization outcome.

The analyses suggest the optimal number of maintenance teams is maintenance strategy dependent. They also reveal, returning components into operation during maintenance suspensions improves system performance. This improvement is attributable to the increased availability of the components culminating in a lower EENS. The exception is the case when PM is initiated only while components are not in operation. In this regime, the initiation of a component's PM is determined by the failure characteristics of other components. Therefore, when the component is returned into operation, its PM resumes only on the occurrence of another shut-down event. The likelihood that the component fails in this interval is higher than in the other regimes, due to the longer wait times. The result is, a fewer PM actions, more failures, longer component downtimes, and a higher EENS. These consequences are minimized by keeping the component out of operation until PM resumes. However, in both cases, initiating preventive maintenance only while components are not in operation yields the best performance.

The range of k_f used in the sensitivity analyses is a little unrealistic for practical applications. The range of interest, therefore, is conservatively chosen to be $0 \leqslant k_f \leqslant 2$, depicting an inflation of -100% to $+100\%$. In this range, the optimal maintenance strategy is unaffected by variations in cost levels, though the number of teams required

for optimal performance varies with the cost of electricity. The following, therefore, is recommended for the hydroelectric power plant.

1. PM should be carried out only when a component is not in operation. That is, it should coincide with a shut-down event that renders the component inactive.

2. Components should be kept out of operation during maintenance interruptions.

3. At the current cost levels, five maintenance teams, employing a shared maintenance strategy, are required for optimal performance. This, however, should be scaled down to 3, 2, 1, and 0, when the cost of electricity deflates by 50%, 60%, 90%, and 100%, respectively, as depicted in Figure 6.10.

4. As evidenced in Figures 6.8 and 6.9, preventive maintenance does not quite improve the overall performance of the system, contrary to anticipations. This, as explained earlier, could be due to subjecting components exhibiting exponential failure characteristics to needless preventive maintenance. It is anticipated that if preventive maintenance is not carried out on these components, additional gains could be made from improved plant availability and reduced maintenance costs. This hypothesis is tested, and as expected, results in an output gain of 1.82% and a corresponding system loss reduction by 7%. Preventive maintenance, therefore, should not be carried out on the breakers, synchronizer, and transformers.

6.4.2 Case-Study 2: The IEEE-24 Bus Reliability Test System

In this case-study, a more realistic system is considered, in order to showcase the applicability of the proposed approach to systems of practical nature. Shown in Figure 6.13 is the single-line diagram of the IEEE-24 bus one-area test system, adapted from [107]. It is composed of 24 buses, 34 power lines, 10 generation stations, and 17 load points. Its total generating capacity is 3405MW with a varying load which annual peak is 2850MW. The total generating capacity and load are distributed across the network as described in [112]. The buses are assumed perfectly reliable and the transmission lines, binary-state. The failure and repair characteristics of the transmission lines in [112] are retained but a few other properties are modified, to make the system more realistic and compatible with the proposed approach. These modifications are summarised as follows.

- Multiple generation units at a bus are represented by a single unit with a generating capacity equivalent to the sum of the generating capacities of the units.

- To make the network more sensitive to the unavailability of transmission lines and generation units, the maximum transmission capacities of the former and

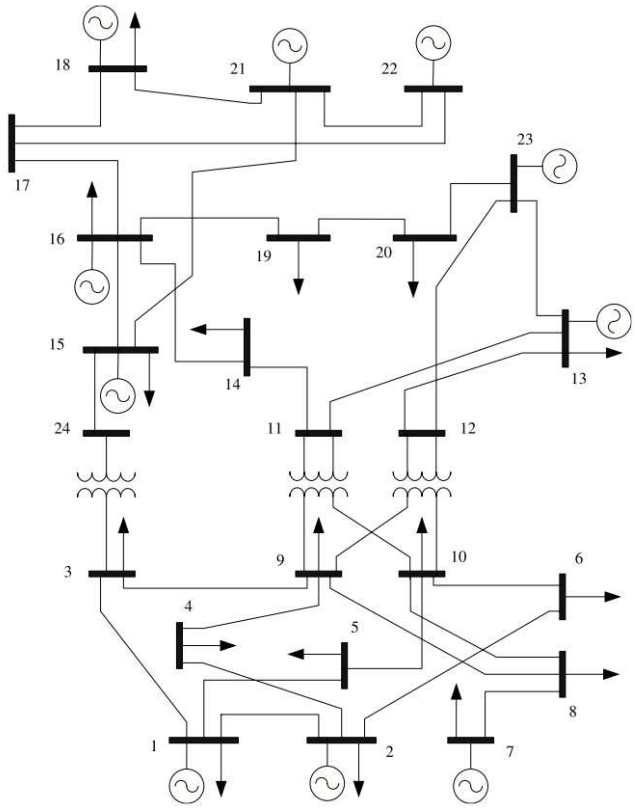


Figure 6.13: Single-line diagram of the IEEE-24 bus Reliability Test System.

Table 6.7: Maintenance data for generation units.

Gen. Type	Bus Number	Spare Usage Prob.		PM		Transition Distribution Parameters				
		CM	PM	Interval	Duration	1-2	2-1	2-3	1-3	3-1
1	22	0.7	0.9	1200	U(156,180)				Wb(2234,2)	Exp(20)
2	1 & 2	0.9,0.25	0.9	1200	U(60,66)	Exp(980)	Exp(20)	Wb(1106,2.3)	Wb(2212,2)	Exp(40)
3	7	0.8,0.4	0.9	1200	U(60,66)	Exp(600)	Exp(25)	Wb(677,2.3)	Wb(1354,2)	Exp(50)
4	15,16 & 23	0.8,0.3	0.9	1000	U(81,87)	Exp(480)	Exp(20)	Wb(542,2.3)	Wb(1083,2)	Exp(40)
5	13	1.0,0.5	0.9	1000	U(102,108)	Exp(575)	Exp(50)	Wb(649,2.3)	Wb(1298,2)	Exp(100)
6	18 & 21	1,0.6	0.9	1000	U(123,129)	Exp(550)	Exp(75)	Wb(621,2.3)	Wb(1241,2)	Exp(150)

minimum allowable loads of the latter are considered in the analysis. These capacities and limits, are respectively given in [112] and [107]. Please note that the minimum load for the unit at bus 22 is set to 25MW instead of the 300MW suggested in [107]. The reason for this is that its contribution to the total load when every component works correctly is only about 37.5MW. A minimum allowable load of 300MW, therefore, would mean it operates only on the failure of another unit. This, in other words, reduces the unit to cold standby, thereby defeating the intention of making every component useful to the system throughout the mission.

- The buses are assigned maximum capacities according to the following rules.

  1. For load and generation buses, the maximum capacity is arbitrarily set to 3 times the capacity of the generation unit or load.

Table 6.8: Maintenance costs for generation units.

Gen. Type	CM		PM	
	CS	CPHM	CS	CPHM
1	180	20	108	12
2	180	20	108	12
3	180	20	108	12
4	200	25	120	15
5	280	40	168	24
6	300	50	180	30

2. For buses with both a generation unit and load, the capacity is set to 3 times the generating capacity or load, whichever is greater.
3. For all other buses, the capacity is set to 3 times the maximum of the capacities of the buses they are connected to.

- Each generation unit, with the exception of the unit at bus 22, is assumed to exist at three possible distinct output levels; 100%, 50%, and 0% of its rated capacity. Unit 22 operates at only two levels; 100% and 0% rated capacity.

6.4.2.1 Maintenance Information

The failure times of the transmission lines are exponentially distributed. As a consequence, they undergo CM only, with an assumed 0.9 likelihood of spares being used. It is also assumed the maintenance crew are able to carry with them these spares. The maintenance of the lines, therefore, is immune to delays in the availability of spares.

The generation units, on the other hand, undergo both PM and CM, and are susceptible to all the operational dynamics described in Section 6.2. Table 6.7 contains their failure and maintenance parameters, where states 1, 2, and 3, respectively represent nominal performance, partial, and complete failure. Their replacement probability during CM is represented by a pair which elements respectively define the probabilities associated with states 3 and 2. Where applicable, the diagnosis and CM durations have the same distribution, with means in the ratio, $1:4$. For instance, the transition of the unit at bus 13 from state 3 to 1, denoting repairs from complete failure, is exponentially distributed with mean 100. Therefore, the diagnosis and corrective maintenance durations are also exponentially distributed with means 20 and 80 respectively. All transition times are in hours and k_i for generation units is conservatively assumed to be 0.3. Also note that the data presented in Table 6.7 are for illustrative purposes only.

6.4.2.2 Maintenance Grouping and Costs

The network components are arranged into three maintenance groups, and each group maintained by a separate maintenance company. The transmission lines above buses 11, 12, and 24 make maintenance group 1, the remaining lines, group 2, and the generation units constitute group 3. Each maintenance team in groups 1 and 2 is paid a fixed £5 per hour and a fixed £100 per successful maintenance action. Teams in group 3 earn £8 every hour and £120 for every successful maintenance action. The cost of one transmission line spare is averaged at £150, the cost per hour of transmission line maintenance, at £15, and the cost levels for the generation units, as defined in Table 6.8. The operator imposes the total number of maintenance teams to not exceed 16.

6.4.2.3 Objective

In the current maintenance strategy, hereafter referred to as the base strategy, corrective and preventive maintenance can be initiated at any time, subject to the availability of maintenance teams. For an annual load cycle of 8736 hours (see [112]) and £100 per MWh of electricity consumed, the optimal maintenance team size is determined for this strategy and its effectiveness compared with three complex strategies. The base strategy, for simplicity, is labelled strategy 1, and the other strategies, as outlined thus.

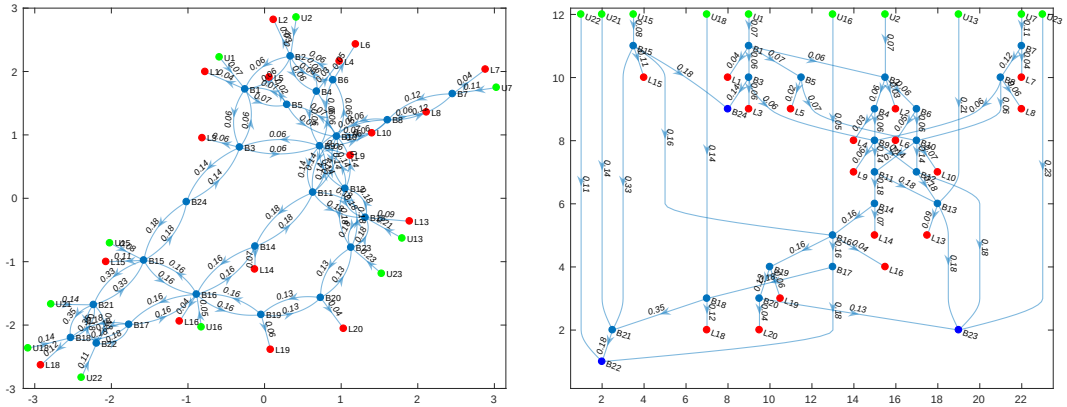
- Strategy 2: PM and CM of partially failed units only when in shut-down.
- Strategy 3: PM and CM of partially failed generation units only when system performance is nominal.
- Strategy 4: PM of generation units only when system performance is nominal but CM of partially failed units can be carried out at any time.

Each maintenance strategy is computed for the case when the units;

- (a) are kept out of operation during maintenance suspensions
- (b) are returned into operation during maintenance suspensions

6.4.2.4 System modelling

Since the goal is to identify the optimal maintenance strategy, a DC flow analysis, using the procedure proposed in Chapter 3, is employed to compute the system reliability and performance indices. The buses, generation units, and load points are modeled as nodes, while the transmission lines are modeled as edges, in the system graph model. In this case study, the edge attribute of the transmission lines has been retained, to keep the number of nodes moderate and improve performance. Consequently, the vector of maximum edge capacities is modified after every transition involving a transmission



(a) Both reciprocal edges shown (b) Only one reciprocal edge shown

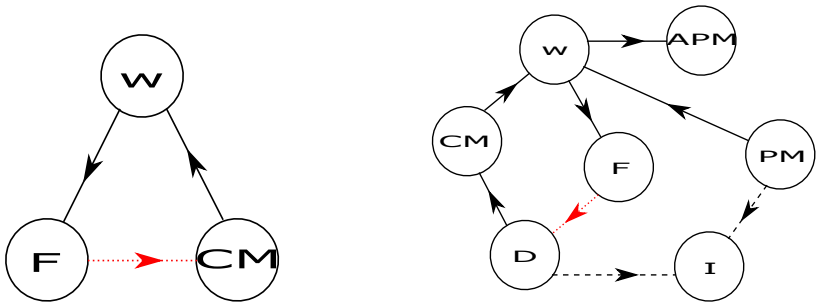
Figure 6.14: System graph model.

line, and both this vector and the vector of node capacities are required for system flow calculation. Figure 6.14 (a) shows the graph model of the system, where Un and Ln respectively denote the generation unit and load point at bus n. Figure 6.14 (b) is the same graph but with only one edge of each reciprocal pair (see Chapter 3) shown, for clarity. In both cases, the number along each edge defines the maximum flow along that edge, as a fraction of the annual peak load.

The effective EENS of the system (given the multiple load points) could be computed as proposed in Section 6.2. However, the computation is rendered less complicated by representing the global system output by a virtual node which flow is the sum of the flows through all 17 load points. The flow history of this virtual node is recorded during simulation, and subsequently used to compute the effective EENS, instead of all 17 nodes. Being mindful of the computational demand of simulation algorithms, a smart procedure is employed to treat the variable demand on the system. Recall the objective of system reliability analysis is to determine the maximum achievable system performance as a consequence of component failure and maintenance. For this reason, the instantaneous system performance, $Y(t)$, is obtained with the assumption that the demand is fixed at its peak annual value. Under this assumption, however, the system is no longer strictly demand-driven (since the actual demand varies with time), and $Y(t)$ has to be normalized to make it compatible with Equations 6.1 and 6.2. The normalization entails expressing $Y(t)$ as a function of the same time-step as the instantaneous demand, $Y_d(t)$, such that they both have equal lengths, and applying,

$$Y(t) = min\{Y(t), Y_d(t)\} \tag{6.17}$$

Normally, variable demand is treated by performing the simulation with respect to the time-step defined by the demand and the events generated by component failures and maintenance. It is, therefore, easy to deduce the computational efficiency of the



(a) Transmission lines (b) Generation unit at bus 22

Figure 6.15: Simplified multi-state model for binary-state components.

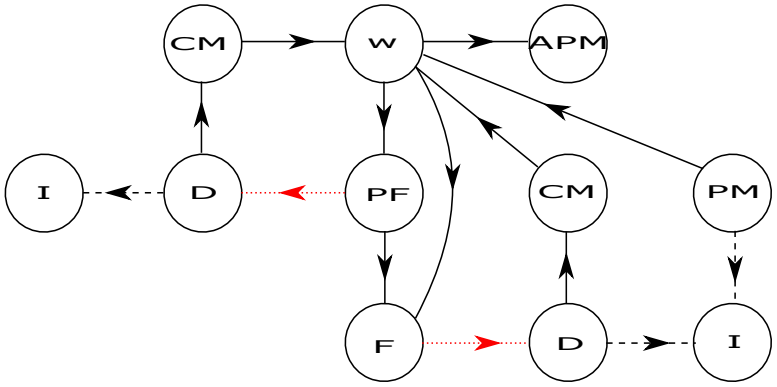


Figure 6.16: Simplified multi-state model for multi-state generation units.

procedure employed here, relative to the widely practised. The procedure is correct for all single-load-point systems, as well as multiple-load-point systems where the quantity of interest is the total output, and not the output through the individual load points.

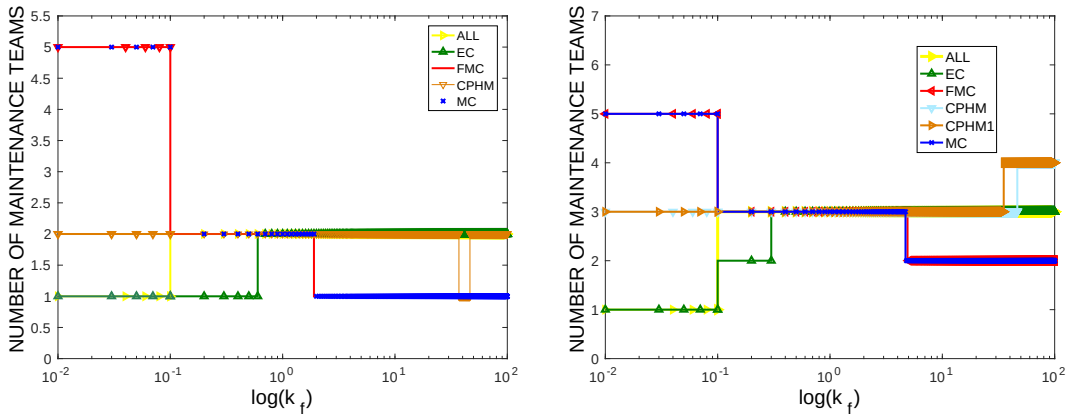
To derive the set, $\mathbb{N}$, of possible maintenance team combinations, the possibility of a 0 maintenance team in any of the maintenance groups is ignored. This is due to the fact that it is known already (from the previous case study) non-repairable maintenance strategies to be grossly inefficient. Recall also that maintenance groups 1 and 2 are composed of equal number of components with the same failure and repair characteristics. They, therefore, have the same optimal maintenance team size. Given these constraints and the upper bound imposed by the operator on the total number of maintenance teams, $\mathbb{N}$ would contain 50 maintenance team combinations.

6.4.2.5 Component Modelling

Figures 6.15 and 6.16 are the system's simplified component models, showing only the required transitions, as discussed in Section 6.3.5. Since the transmission lines are not susceptible to maintenance interruptions, their failure diagnosis and actual repair have been collectively represented by the CM state. This, however, implies the number of spares used cannot be directly obtained from the simulation, as spares used are

Table 6.9: Optimal System Loss as a function of maintenance strategy.

Strategy		EENS(%)	L($£10^6$)	Optimal number of teams		
				Group 1	Group 2	Group 3
1	a	0.3940	6.63	1	1	3
	b	0.2468	4.47	2	2	3
2	a	2.4218	37.66	1	1	4
	b	2.4780	38.68	3	3	4
3	a	1.3592	21.36	1	1	3
	b	1.5049	23.65	1	1	4
4	a	0.3373	5.90	1	1	5
	b	0.2128	3.95	2	2	3



(a) Groups 1 and 2

(b) Group 3

Figure 6.17: Optimal maintenance team sensitivity to cost levels.

accounted for only if the component enters Diagnosis or PM state (see Algorithm 6). The total spares used, therefore, is obtained from the product of the spares usage probability and the number of CM to W transitions. Please note the models in Figures 6.15 and 6.16 are based on the assumption that components are kept out of operation during maintenance suspensions. Those for the case when components are returned into operation can be easily deduced from Figures 6.2-6.4. It is also worthwhile noting that, the simplified component models for regimes 1 to 3 of Section 6.2.3 are equivalent.

6.4.2.6 Results and Discussions

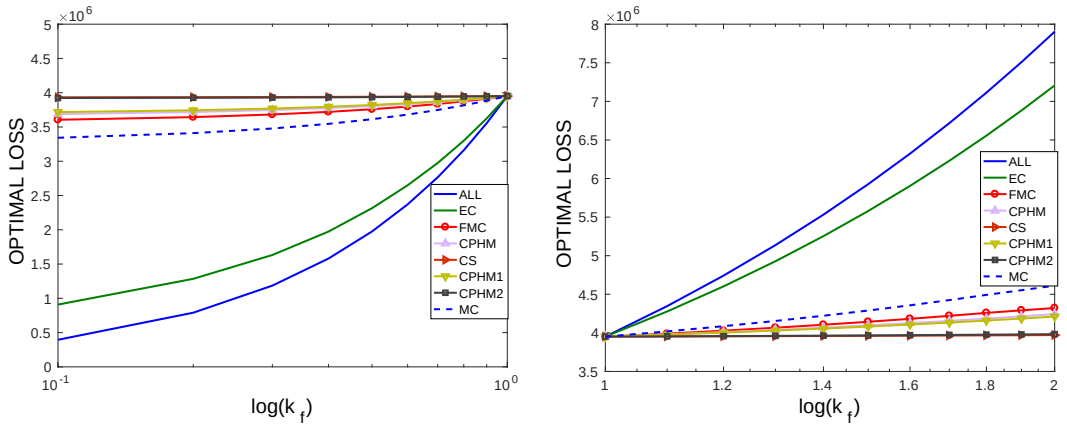
The system was analysed on the same computer used for the previous case study, and the outcome is summarized in Table 6.9. The table provides the EENS as a percentage of the total expected output, the expected loss, and the optimal maintenance team combination for each strategy. Each sample of a candidate solution took an average of $0.8s$, using 10 MATLAB workers. Given the large number of candidate solutions, the number of samples per candidate solution was set to 500. The sensitivity of the optimal solution to the costs considered in the previous case study and a few other costs, was also investigated. The additional costs considered are as follows.

- Cost per hour of CM and cost per CM call (CPHM1).

- Cost per hour of PM and cost per PM call (CPHM2).
- Total maintenance cost (MC); a combination of FMC, CPHM1, and CPHM2.
- All costs relevant to the system loss function (ALL).

Deducing from the data in Table 6.9, the optimal maintenance strategy is strategy 4 (b). In this strategy, the corrective maintenance of partially failed generation units can be initiated at any time but preventive maintenance, only when system performance is nominal, with components returned into operation during maintenance suspensions. Postponing both corrective and preventive maintenance until component shut-down appears to be the most inefficient, contrary to what obtained in the previous case study. This observation reiterates the point that the optimality of a given maintenance strategy depends on specific properties of the system. For $0 \leqslant k_f \leqslant 100$, strategy 4 (b) remains optimal, but the optimal number of maintenance teams varies as depicted by Figure 6.17. It should be noted that cost parameters with no effect on the optimal number of maintenance teams have been left out in Figures 6.17 (a) and 6.17 (b). Given maintenance groups 1 and 2 are made up of the transmission lines only (which do not undergo PM), CPHM and CPHM1 are equivalent, explaining the absence of CPHM1 and CPHM2 in Figure 6.17 (a). A notable conclusion drawn from Figure 6.17 is that the optimal number of maintenance teams is most affected by the cost of electricity (EC) and the fixed cost per maintenance team (FMC). It is also easily deducible that the number of teams required for optimality reduces and increases with reduction in EC and FMC, respectively, both observations conforming to common reasoning.

Figure 6.18 shows the variation in system loss with changes in cost levels in the range, $0 \leqslant k_f \leqslant 2$. For clarity, system response over the ranges $0 \leqslant k_f \leqslant 1$ and $1 \leqslant k_f \leqslant 2$ has been presented separately in Figures 6.18 (a) and 6.18 (b), respectively. With $k_f = 1$ as reference, Figure 6.18 (a) defines the sensitivity of the total system loss to cost reductions and Figure 6.18 (b), to cost increments. In both cases, the cost of electricity and the overall maintenance cost impact system loss the most. However, the system shows very little sensitivity to both the cost of spares and the cost per hour of PM action, suggesting a few PM actions and low spares usage. The low system loss sensitivity to CPHM2 is explained by the fact that only 10 of the 44 system components undergo PM. Given strategy 4 imposes PM be initiated only if system performance is nominal, a good number of these components fail before their PM commences.



(a) Cost reduction

(b) Cost increment

Figure 6.18: System loss sensitivity to cost levels.

6.5 Chapter Summary

It is realistic to think that increasing the number of maintenance teams improves the performance and reliability of a multi-component system. However, a threshold exists exceeding which no gains are realised. Rather, it results in increased operational costs, borne from the imbalance between income and expenditure. This threshold varies with the maintenance strategy, the input costs to the system's cost model, the topology of the system, and the non-topological functional relationships between its components.

In this chapter, a maintenance strategy optimization framework, aiding proper maintenance scheduling and robust maintenance decisions has been presented. Applicable to binary and multi-state systems, the framework proposes a multi-state model to define the behaviour of components under various maintenance strategies. A nonsystem-specific, event-driven Monte Carlo simulation, based on the load-flow approach proposed in Chapter 3 is employed to replicate the operation of the system. This simulation algorithm, together with the multi-state component model, enhances the implementation of complex maintenance strategies. For instance, a component may belong to two maintenance groups practising dedicated and shared maintenance, respectively. There could also exist multiple maintenance groups, with some initiating maintenance promptly and others only during a shut-down event or at the attainment of nominal system performance. Many more contrasting combinations of regimes are possible, without the need to modify the simulation algorithm. The framework is also built on a cost model structured to allow the sensitivity analysis of the optimal solution from a single reliability evaluation. These attributes render it novel, efficient, and generally applicable to power & water distribution networks, as well as other flow-based systems.

The framework has been successfully used to optimize the maintenance strategies for two realistic power systems, obtaining insightful information on their maintenance.

The relationship derived between the optimal number of maintenance teams and the cost of electricity, for instance, is a very useful tool, given a volatile electricity market. The framework, therefore, can shape the quality of maintenance-related decisions, even in the presence of external dynamics.

Chapter 7

An Extended Survival Signature Approach for Dependent Failures

7.1 Introduction

Dependent failures impose severe consequences on a complex system's reliability and overall performance, and a realistic assessment requires an adequate consideration of these failures. System survival signature opens up a new and efficient way to compute a system's reliability, given its ability to segregate the structural from the probabilistic attributes of the system. Consequently, it outperforms the well-known system reliability evaluation techniques (like the ones proposed in Chapters 3 and 4), when solicited for problems like maintenance optimization, requiring repetitive system evaluations. The survival signature, however, is premised on the statistical independence between component failure times and more generally on the theory of weak exchangeability, for dependent component failures. The assumption of independence is flawed for most realistic engineering systems whilst the latter entails the painstaking and sometimes impossible task of deriving the joint survival function of the system components. This chapter, therefore, proposes a novel, generally applicable, and efficient Monte Carlo simulation approach that allows the survival signature to be intuitively used for the reliability evaluation of systems susceptible to induced failures. Multiple component failure modes, as well, are considered, and sensitivities are analysed to identify the most critical Common-Cause Group to the survivability of the system. Examples, which results obtained are under consideration for publication in the *Journal of Risk and Reliability*, demonstrate the superiority of the approach.

The next section of this chapter presents a general overview of the proposed approach. Section 7.3 contains the detailed modelling procedure and the Monte Carlo simulation algorithm. The two examples illustrating the applicability of the approach are presented in Section 7.4, followed by a summary of the chapter, in Section 7.5.

7.2 Overview of Proposed Approach

The existing survival signature-based modelling formalism is harnessed to propose a more realistic approach to system reliability analysis. In the proposed approach, the survival signature of the system is obtained prior to system analysis, using its topological attributes only. An efficient event-driven Monte Carlo simulation is then invoked to recreate the failure of the components and propagate the ensuing dependencies, where necessary. The number of operating components is determined for each component type, with the corresponding system survival signature directly read off from a predefined register and saved as a function of time. These stored values are later used to compute the time-dependent reliability of the system, using basic probabilistic principles.

Since the survival signature of a system is fixed so long as its topology does not change, the proposed approach makes an efficient alternative. Its efficiency particularly stands out in maintenance optimization problems, sensitivity & uncertainty analyses, and other problems requiring multiple system reliability evaluations. Also, because it is simulation-based, it can accommodate any component failure time distribution type, including user-defined distributions. In summary, the proposed approach inherits the desirable attributes of both the survival signature-based and Monte Carlo simulation-based approaches. It is, to the best of my knowledge, the first documented extension of the survival signature-based approach to the complete system level reliability evaluation of complex systems susceptible to both common-cause and cascading failures.

7.3 Modelling & Simulating the System

Consider the system described in Section 2.5.1, and suppose the random failure of a component of type k may trigger the failure of one or more components. In addition, let one or more components have multiple total failure modes, which failure modes in turn may have different effects on the system. By total failure I mean, the component is completely failed and its output/structure function is 0. The component, in other words, is still deemed binary-state. Suppose also that the system is not only susceptible to cascading failures emanating from within its boundaries but to cascading failures triggered by external factors, as well. Clearly, the existing survival signature-based reliability evaluation approaches are inadequate for such a system. This section provides a detailed description of the modelling approach for such systems.

7.3.1 Components with Multiple Failure Modes

The survival signature is suited to binary-state components and systems. Therefore, when the system being modelled also contains components with multiple failure modes, the analyst would need to make these compatible with the signature-based approach.

Consider a component with two failure modes, and which occurrence times follow the CDF, $F_1(t)$ and $F_2(t)$, respectively. If these failure modes have the same effect on the system, then, they can be merged. Two total failure modes have the same effect if they do not trigger dependent failures or if they have an equal likelihood of affecting the same set of components. The effective CDF, $F_i(t)$, of the component is computed from the probability, $P(min(T_1, T_2) \leqslant t)$, where T_1 and T_2 are the random occurrence times of failure modes 1 and 2, respectively. This follows from the reasoning that the component is failed on the occurrence of any of the failure modes. The resulting probability could also be viewed as the complement of the probability that none of the failure modes occurs, yielding, $F_i(t) = 1 - [1 - F_1(t)][1 - F_2(t)]$. Generally, the effective CDF, $F_i(t)$, of an n failure mode component is given by $F_i(t) = 1 - \prod_{l=1}^{n}[1 - F_l(t)]$, so long as the failure modes are total and impose the same effect on the system.

There are times when a set of component failure modes do not satisfy the condition for merging. Currently, such a scenario cannot be solved analytically, and I will, therefore, not be bothered about computing the effective CDF of the component. Instead, I will propose a set of procedures to segregate the component into several binary-state elements, which then can be easily implemented by a Monte Carlo simulation algorithm. It should be noted that this segregation is only required to enhance the intuitive representation of the inter-component dependencies and ensure a simplified simulation algorithm. The system, otherwise, could still be analysed, only the sampling algorithm proposed in Chapter 3 and the dependency matrices proposed in Chapters 4 and 6 would be required, complicating an otherwise simple solution procedure.

An n failure mode component, i, is segregated by redefining its state-space to contain the working state (conservatively assumed to be state 1), and one of the failure modes (assumed to be state 2). The remaining $n - 1$ failure modes (assumed to be state 3 to state n) are each then assigned to a virtual binary-state node. Component i retains the failure time distribution to state 2 while the virtual nodes inherit the failure time distributions to their respective failure modes. The virtual nodes, as their name implies, are not really a part of the system, and should, therefore, be considered external factors/nodes. External nodes are not considered when deriving the survival signature set, $\boldsymbol{S}_\tau$, of the system, which is why their numbering starts from $M + 1$, M being the number of system components. Since in practice, the virtual nodes, together with the parent node, i, represent the same component, the failure of any of these nodes denotes the failure of the component. Hence, each virtual node is a dual of the parent node.

When a node fails, its duals can no longer affect the system, since the failure modes of a component are mutually exclusive and in this chapter, non-repairable. Consequently, the system simulation algorithm should be equipped with a special routine to ensure affected dual nodes are removed, following a failure event. For this, an efficient recursive algorithm is proposed in this chapter. The algorithm takes the current values of $\underline{\boldsymbol{x}}'$, $\boldsymbol{\rho}^{\{k\}}$ for all $k \in \{1, 2, ..., K\}$, the set, $\mathbb{F}$, of failed components, and the set, $\mathbb{D}_i$, of duals of

Algorithm 8 Procedure for removing dual nodes.

Require: $\underline{\boldsymbol{x}}'$, $\mathbb{F}$, $\boldsymbol{\rho}$, $\mathbb{D}$, i

1: **function** RemoveDualNodes($\underline{\boldsymbol{x}}'$, $\mathbb{F}$, $\boldsymbol{\rho}$, $\mathbb{D}$, i)
2: $\quad\mathbb{D}_i \leftarrow \mathbb{D}_i - \mathbb{F}$ ▷ Remove failed nodes
3: $\quad$**if** $\mathbb{D}_i \leftarrow \varnothing$ **then**
4: $\quad\quad$go to line 13 ▷ Exit algorithm if i has no duals
5: $\quad$**end if**
6: $\quad$**for** $j \in \mathbb{D}_i$ **do** ▷ Loop over dual nodes
7: $\quad\quad k \leftarrow$component type of node j
8: $\quad\quad\boldsymbol{\rho}^{\{k\}} \leftarrow \boldsymbol{\rho}^{\{k\}} - j$ ▷ Remove j from set
9: $\quad\quad(\underline{\boldsymbol{x}}', k) \leftarrow \mid \boldsymbol{\rho}^{\{k\}} \mid$ ▷ Update state vector
10: $\quad\quad\mathbb{F} = \mathbb{F} \cup j$ ▷ Update failed nodes/components
11: $\quad\quad(\underline{\boldsymbol{x}}', \mathbb{F}, \boldsymbol{\rho}) \leftarrow$RemoveDualNodes($\underline{\boldsymbol{x}}', ..., j$)
12: $\quad$**end for**
13: $\quad$**return** ($\underline{\boldsymbol{x}}'$, $\mathbb{F}$, $\boldsymbol{\rho}$)
14: **end function**

component i for all $i \in \{1, 2, ..., M + M''\}$, returning $\underline{\boldsymbol{x}}'$, $\boldsymbol{\rho}^{\{k\}}$, and $\mathbb{F}$, where M'' is the number of external nodes. Following the failure of a component, the algorithm first removes all its duals that are not in operation, from $\mathbb{D}_i$. The component type, k, of the first active dual is determined, following which it is removed from the set of components, $\boldsymbol{\rho}^{\{k\}}$, in that group, and the k^{th} element of the modified system state vector, $\underline{\boldsymbol{x}}'$, replaced with the cardinality of $\boldsymbol{\rho}^{\{k\}}$, which in other words is written as $(\underline{\boldsymbol{x}}', k) =\mid \boldsymbol{\rho}^{\{k\}} \mid$. Due to the possibility of a dual node possessing its own duals, the algorithm is recursively applied to the node, as highlighted on line 11 of Algorithm 8. The sequence is repeated for the remaining active duals, adding each to set $\mathbb{F}$ before moving on to the next. Algorithm 8 summarises the procedure for removing the duals of component i, following its failure. In the algorithm and the remainder of this chapter, $\mathbb{D}$ denotes the global set of $\mathbb{D}_i \forall i \in \{1, 2, ..., M + M''\}$, such that $\mathbb{D} = \{\mathbb{D}_1, \mathbb{D}_2, ..., \mathbb{D}_{M+M''}\}$. Similarly, $\boldsymbol{\rho}$ denotes the global set of $\boldsymbol{\rho}^{\{k\}} \forall k \in \{1, 2, ..., K\}$, such that $\boldsymbol{\rho} = \{\boldsymbol{\rho}^{\{1\}}, \boldsymbol{\rho}^{\{2\}}, ..., \boldsymbol{\rho}^{\{K\}}\}$.

7.3.2 Cascading Failure Modelling and Propagation

The cascading dependency between components is defined by the cascading matrix, $\mathbb{C}$. The cascading matrix, which can be a sparse matrix, is an $(M + M'')$ order square matrix which elements denote whether or not the failure of a component can trigger the almost instantaneous failure of another component. The element in row i and column j of the matrix is assigned the value 1 if the failure of component i can induce failures (in the cascading failure sense) in component j, and 0, otherwise. Therefore, the set, $\mathbb{I}_i$, of components which failure is induced by component i is given by the column indices of the non-zero elements of row i of $\mathbb{C}$.

Algorithm 9 Procedure for forcing cascading failures.

Require: $\underline{\boldsymbol{x}}'$, $\mathbb{F}$, $\boldsymbol{\rho}$, $\mathbb{D}$, $\mathbb{C}$, i

1: **function** CascadeFailure($\underline{\boldsymbol{x}}'$, $\mathbb{F}$, $\boldsymbol{\rho}$, $\mathbb{D}$, $\mathbb{C}$, i)
2: $\mathbb{I}_i \leftarrow$induced components obtained from $\mathbb{C}$
3: $\mathbb{I}_i \leftarrow \mathbb{I}_i - \mathbb{F}$ ▷ Remove failed nodes
4: **if** $\mathbb{I}_i \leftarrow \varnothing$ **then**
5: go to line 15 ▷ Exit if i cannot induce failure
6: **end if**
7: **for** $j \in \mathbb{I}_i$ **do** ▷ Loop over induced components
8: $k \leftarrow$component type of node j
9: $\boldsymbol{\rho}^{\{k\}} \leftarrow \boldsymbol{\rho}^{\{k\}} - j$ ▷ Remove j from set
10: $(\underline{\boldsymbol{x}}', k) \leftarrow | \boldsymbol{\rho}^{\{k\}} |$ ▷ Update state vector
11: $\mathbb{F} = \mathbb{F} \cup j$ ▷ Update failed nodes/components
12: $(\underline{\boldsymbol{x}}', \mathbb{F}, \boldsymbol{\rho}) \leftarrow$RemoveDualNodes($\underline{\boldsymbol{x}}', ..., j$)
13: $(\underline{\boldsymbol{x}}', \mathbb{F}, \boldsymbol{\rho}) \leftarrow$CascadeFailure($\underline{\boldsymbol{x}}', ..., j$)
14: **end for**
15: **return** ($\underline{\boldsymbol{x}}'$, $\mathbb{F}$, $\boldsymbol{\rho}$)
16: **end function**

To account for these cascading dependencies, a second recursive algorithm is proposed, to propagate their effects across the system during simulation. The algorithm takes in the same input set required by Algorithm 8 in addition to the cascade matrix and returns $\underline{\boldsymbol{x}}'$, $\boldsymbol{\rho}^{\{k\}}$, and $\mathbb{F}$. Following the failure of a component, the algorithm first deduces the possible set of components that can be affected. From this set, currently inactive components are removed, and the rest of the procedure is similar to what obtains in Algorithm 8. Since an induced component can also induce failures in other components, the algorithm recursively calls itself, this time, to propagate any failures the induced may induce. The procedure is summarised by Algorithm 9.

7.3.3 CCF Modelling and Propagation

For non-repairable binary-state systems, a CCG is characterised by a set of probabilities. This set defines the likelihood of a given number of components being involved in any random failure event affecting the group.

Let the CCF probability for component type k be defined by $\boldsymbol{\theta}^{\{k\}}$, such that $\boldsymbol{\theta}^{\{k\}} = \{\theta_r^{\{k\}}\}^{M_k} = \{\theta_1^{\{k\}}, \theta_2^{\{k\}}, ..., \theta_{M_k}^{\{k\}}\}$, r being the total number of components affected by the failure event, and $\sum_{r=1}^{M_k} \theta_r^{\{k\}} = 1$. In effect, $\theta_r^{\{k\}}$ denotes the probability of an additional $r-1$ components failing, following the failure of a type k component, in conformity with the α-factor model. A key requirement, therefore, is that CCF probabilities are expressed according to this model. Probabilities expressed according to the Multiple

Greek Letter model would need to be converted as outlined in Ref. [104]

$$\mathbb{H} = \{H_{kr}\}^{K \times max\{\boldsymbol{\xi}\}} \mid H_{kr} = \begin{cases} \theta_r^{\{k\}} & \text{If } r \leqslant M_k \\ 0 & \text{otherwise} \end{cases} \tag{7.1}$$

In the most general sense, the notation established in the preceding paragraph could as well be used for component types immune to CCF. For this special case, $\theta_1^{\{k\}} = 1$ and $\theta_r^{\{k\}} = 0$ for all $r > 1$, which by definition, means, the probability of no additional component failing, following the failure of a type k component is 1. Leaning on this fact, the CCF matrix, $\mathbb{H}$, is introduced to define the CCF characteristics of a system with a mix of component types susceptible and immune to CCF. $\mathbb{H}$ is a $K \times max\{\boldsymbol{\xi}\}$ matrix, where $\boldsymbol{\xi} = \{M_1, M_2, ..., M_{K-1}, M_K\}$ is the set of number of components, $M_k \mid k = 1, 2, ..., K$, in each group. Each row of $\mathbb{H}$, therefore, defines the CCF characteristics of the component type corresponding to the index of that row, as outlined as in Equation 7.1. The attributes of $\mathbb{H}$ impose two constraints;

1. A component can only belong to one CCG. This implies, CCF events in one CCG are independent of the CCF events in other CCGs. They can, however, still induce cascading failures in these CCGs.

2. For a given component type k, all its M_k components must belong to the same CCG. What this means is, no component should be immune to a CCF to which some components of the same type are susceptible. Strictly speaking, in real life, it is unlikely to have a CCF affecting only a fraction of the components of a given type. However, on its unlikely occurrence, I suggest the components be segregated into two new component types, based on their susceptibility to CCF.

These constraints should, therefore, be kept in mind when defining component types. As a rule-of-thumb, every component type should be viewed as a CCG, and defined as such, whether or not it is susceptible to CCF. This, indeed, is logical, since components of the same type have similar characteristics, and would, therefore, be influenced by the same common-cause event.

With the CCF modelling procedure outlined, the remainder of this section details CCF propagation during system simulation. Recalling simulation entails recreating the actual operating principles of a system, a very simple procedure for propagating CCF, following the failure of a component is proposed. When a member of a CCG fails, there could be $0, 1, 2$ up to $M_k - 1$ additional component failures. The total number of component failures is determined by the CCF matrix, as discussed earlier. Therefore, following a component failure, the number of additional components to fail is first deduced. This is achieved by generating a uniform random number, U, between 0 and 1 and comparing it to the cumulative sum, $\mathbb{H}'$, of $\mathbb{H}$. $\mathbb{H}'$ is the cumulative sum of the

Algorithm 10 Procedure for propagating CCF.

Require: $\underline{\boldsymbol{x}}'$, $\boldsymbol{\rho}$, $\mathbb{F}$, $\mathbb{D}$, $\mathbb{H}'$, $\mathbb{C}$, k

1: **function** PROPAGATECCF($\underline{\boldsymbol{x}}'$, $\boldsymbol{\rho}$, $\mathbb{F}$, $\mathbb{D}$, $\mathbb{H}'$, $\mathbb{C}$, k)
2: **if** $H'_{k1} \leftarrow 1$ **or** $\boldsymbol{\rho}^{\{k\}} \leftarrow \varnothing$ **then**
3: go to line 16 ▷ Exit if CCF is not possible
4: **end if**
5: get r ▷ Get number of components to fail
6: **if** $r > 1$ **then** ▷ Multiple components affected
7: $\boldsymbol{Z} \leftarrow$ set of $(r-1)$ components from $\boldsymbol{\rho}^{\{k\}}$
8: $\boldsymbol{\rho}^{\{k\}} \leftarrow \boldsymbol{\rho}^{\{k\}} - \boldsymbol{Z}$ ▷ Remove components
9: $(\underline{\boldsymbol{x}}', k) \leftarrow |\ \boldsymbol{\rho}^{\{k\}}\ |$ ▷ Update state vector
10: $\mathbb{F} = \mathbb{F} \cup \boldsymbol{Z}$ ▷ Update failed nodes
11: **for** $j \in \boldsymbol{Z}$ **do** ▷ Loop over components
12: $(\underline{\boldsymbol{x}}', \mathbb{F}, \boldsymbol{\rho}) \leftarrow$REMOVEDUALNODES$(..., j)$
13: $(\underline{\boldsymbol{x}}', \mathbb{F}, \boldsymbol{\rho}) \leftarrow$CASCADEFAILURE$(\underline{\boldsymbol{x}}', ..., j)$
14: **end for**
15: **end if**
16: **return** $(\underline{\boldsymbol{x}}', \mathbb{F}, \boldsymbol{\rho})$
17: **end function**

elements of $\mathbb{H}$ along each row, from left to right. Thus,

$$\mathbb{H}' = \{H'_{kr}\}^{K \times max\{\boldsymbol{\xi}\}} \mid H'_{kr} = \sum_{r=1}^{r} H_{kr} \tag{7.2}$$

The total number, r, of components involved in the CCF of a type k component is equal to the index of the smallest element of the k^{th} row of matrix $\mathbb{H}'$ greater than or equal to U, expressed as $r = min\{n, n+1, n+2, ..., max\{\boldsymbol{\xi}\}\} \mid H'_{kn} \geqslant U$.

If $r > 1$, $r-1$ components, excluding the one initiating the CCF, are randomly chosen from the set, $\boldsymbol{\rho}^{\{k\}}$, of components of type k. The selected components are those affected by the CCF event. The condition $r = 1$ denotes the scenario when no additional components are affected by the failure of the first component. On the other hand, if the cardinality of $\boldsymbol{\rho}^{\{k\}}$ is less than or equal to $r-1$, all the active type k components would fail. As in Algorithms 8 and 9, each failed component is removed from $\boldsymbol{\rho}^{\{k\}}$, with $\mathbb{F}$ and $\underline{\boldsymbol{x}}'$ updated accordingly. Algorithms 8 and 9 are also applied to the component, to ensure dual nodes and cascading failures are accounted for, respectively. Since all the components affected by the CCF event belong to the same component type (see the assumptions in this section), they can be removed from $\boldsymbol{\rho}^{\{k\}}$ and added to $\mathbb{F}$, each in just one step, without needing a *for* or *while* loop. Algorithm 10 summarises CCF propagation, following the failure of a component. Line 2 of this algorithm checks whether or not the component failure can induce CCF in other components. The condition $H'_{k1} = 1$ means type k components are not susceptible to CCF while $\boldsymbol{\rho}^{\{k\}} = \varnothing$

suggests none of these components is active, explaining why the algorithm is terminated.

7.3.4 The Simulation Algorithm

Prior to simulation, each system component is assigned a positive integer, $i \mid i \in \{1, 2, ..., M\}$, representing its index in the system. The numbering starts with the components that actually make up the topology of the system, ending with external nodes. These components are then segregated into groups, according to their similarities. For the purpose of this work, the words, component and node, will be used interchangeably to refer to any element that effects the operation of the system.

Let $f_k(t)$ denote the common failure time distribution for all components of type k and $\boldsymbol{f}$, the global set containing $f_k(t)$ for all $k \in \{1, 2, ..., K\}$. To prepare the system for simulation, set the initial state vector, $\underline{\boldsymbol{x}}'$, to $\{M_1, M_2, ..., M_{K-1}, M_K\}$ and the set, $\mathbb{F}$, of failed components to $\varnothing$, since all the components are initially working. For the same reason, the survival signature, $\boldsymbol{S}_\tau\left(\underline{\boldsymbol{x}}'\right)$, at the initial time step, $j_0 = 1$, is assigned a value of 1. The set, $\boldsymbol{\rho}$, the CCF matrix, $\mathbb{H}$, the cascade matrix, $\mathbb{C}$, and the global set of duals, $\mathbb{D}$, are also defined. Finally, the mission time, T_m, is divided into equal time steps of magnitude δ, and the survival function, $R(t)$, preallocated as a vector of zeros, with each element corresponding to a time step. The survival function, in other words, is defined as, $R(t) = \{0\}^{\delta_t}$, where δ_t is the number of time steps.

At time $t = 0$, the failure time of each of the $M + M''$ components of the system is sampled from its appropriate distribution and stored in $\boldsymbol{\tau}$. From $\boldsymbol{\tau}$, the next transition time, t, and the component, i, to fail are deduced. t is equivalent to the minimum element of $\boldsymbol{\tau}$ and i, its index in the set. The simulation is then shifted to this time, at which point the number of time steps, j, t represents is computer as $\lceil t/\delta \rceil$. Elements j_0 to j of the survival function, $R\left(t\right)$, are each incremented by $\boldsymbol{S}_\tau\left(\underline{\boldsymbol{x}}'\right)$. The type, k, of the component is determined, the component is removed from $\boldsymbol{\rho}^{\{k\}}$, added to the set of failed components, and $\underline{\boldsymbol{x}}'$ modified to reflect the changes. Where necessary, Algorithms 8, 9, and 10 are invoked to remove dual nodes, cascade failures, and propagate CCF across the system, respectively. The next transition times of the failed components are set to infinity and j_0 set to $j + 1$. Again, the next transition time, t, and component, i, are determined and the cycle restarts. This procedure continues until $t > T_m$ or $\boldsymbol{S}_\tau\left(\underline{\boldsymbol{x}}'\right) = 0$, which ever occurs first. The second condition is satisfied only when the non-repairable system is failed, explaining why the simulation is terminated on its occurrence.

The sequence of events described in the preceding paragraph accounts for only one simulation sample. Since component failures are random in nature, this sequence should be repeated for an appreciable number of times. The effective survival function of the system is obtained by dividing the final value of $R(t)$ by the number of simulation samples, N. It is, however, worthwhile noting that the accuracy of the outcome is influenced by the values of N and δ. A large N and a small δ (relative to the mission time), guar-

Algorithm 11 System simulation procedure.

Require: $\mathbb{H}'$, N, $\boldsymbol{\rho}$, $\boldsymbol{f}$, $\mathbb{C}$, $\mathbb{D}$

1: **function** SIMULATE($\mathbb{H}'$, N, $\boldsymbol{\rho}$, $\boldsymbol{f}$, $\mathbb{C}$, $\mathbb{D}$)
2: $\underline{\boldsymbol{x}}' \leftarrow \{M_1, M_2, ..., M_{K-1}, M_K\}$ ▷ Initialise $\underline{\boldsymbol{x}}$
3: $R(t) \leftarrow \{0\}^{\delta_t}$ ▷ Initialise survival function
4: $\boldsymbol{\tau} \leftarrow \{0\}^{M+M''}$ ▷ Initialise $\boldsymbol{\tau}$
5: $j_0 \leftarrow 1$ ▷ Define initial time step
6: $\mathbb{F} \leftarrow \varnothing$ ▷ Define initial set of failed components
7: $\boldsymbol{S}_\tau(\underline{\boldsymbol{x}}') \leftarrow 1$ ▷ Set survival signature to 1
8: **for** $k \leftarrow 1$ **to** K **do** ▷ Loop over component type
9: $\left(\boldsymbol{\tau}, \boldsymbol{\rho}^{\{k\}}\right) \leftarrow f_k(t)Ⓢ M_k$ ▷ Sample failures
10: **end for**
11: $[t, i] \leftarrow min\{\boldsymbol{\tau}\}$ ▷ Get next failure time and i
12: **while** $t \leqslant T_m$ **and** $\boldsymbol{S}_\tau(\underline{\boldsymbol{x}}') > 0$ **do**
13: $j \leftarrow \lceil t/\delta \rceil$ ▷ Get the number of time steps
14: $(R(t), j_0 \rightarrow j) \leftarrow (R(t), j_0 \rightarrow j) + \boldsymbol{S}_\tau(\underline{\boldsymbol{x}}')$
15: $k \leftarrow$component type of node i
16: $\boldsymbol{\rho}^{\{k\}} \leftarrow \boldsymbol{\rho}^{\{k\}} - i$ ▷ Remove i from set
17: $(\underline{\boldsymbol{x}}', k) \leftarrow \mid \boldsymbol{\rho}^{\{k\}} \mid$ ▷ Update state vector
18: $\mathbb{F} = \mathbb{F} \cup i$ ▷ Update set of failed nodes
19: $(\underline{\boldsymbol{x}}', \mathbb{F}, \boldsymbol{\rho}) \leftarrow$REMOVEDUALNODES$(\underline{\boldsymbol{x}}', ..., i)$
20: $(\underline{\boldsymbol{x}}', \mathbb{F}, \boldsymbol{\rho}) \leftarrow$CASCADEFAILURE$(\underline{\boldsymbol{x}}', ..., i)$
21: $(\underline{\boldsymbol{x}}', \mathbb{F}, \boldsymbol{\rho}) \leftarrow$PROPAGATECCF$(\underline{\boldsymbol{x}}', ..., k)$
22: $(\boldsymbol{\tau}, \mathbb{F}) \leftarrow \infty$ ▷ Update transition times
23: $j_0 \leftarrow j + 1$ ▷ Set next initial time step
24: $[t, i] \leftarrow min\{\boldsymbol{\tau}\}$ ▷ Get next failure time and i
25: **end while**
26: **if** $j_0 \leqslant n$ **and** $\boldsymbol{S}_\tau(\underline{\boldsymbol{x}}') > 0$ **then**
27: $(R(t), j_0 \rightarrow n) \leftarrow (R(t), j_0 \rightarrow n) + \boldsymbol{S}_\tau(\underline{\boldsymbol{x}}')$
28: **end if**
29: Repeat lines 5 to 28 N times
30: $R(t) = \frac{R(t)}{N}$
31: **return** $(R(t))$
32: **end function**

antee an accurate $R(t)$. Algorithm 11 summarises the simulation procedure, which is different from the Algorithms proposed by Patelli and Feng [111], as the proposed procedure considers dependencies and multiple failure modes. The block of code between lines 26 and 28 updates the survival function after the last component failure. The notation, $f_k(t)Ⓢ M_k$, on line 9 denotes M_k random failure times sampled from $f_k(t)$.

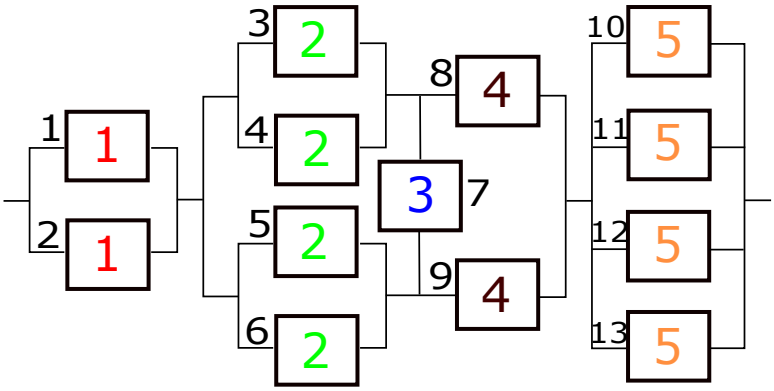


Figure 7.1: An arbitrary multi-component complex network.

7.3.5 Sensitivity Analysis

Sensitivity analysis is the study of how the variation in the output of a mathematical model is apportioned to variations in its inputs [14]. In a complex system with multiple CCGs, each CCG may have a unique effect on the system's survivability.

Consider a complex system with K component types and its CCG matrix, $\mathbb{H}$. The relative influence of the various CCGs on the reliability of the system is a vital decision-support information [77]. The computation of this relative sensitivity entails analysing the system with only one CCG active, for all the CCGs. To compute the effect of CCG i, H_{k1} is set to 1 and $H_{k2}, H_{k3}, ..., H_{kM_k}$ to 0, for all $k \neq i$. This is equivalent to replacing all rows other than i, of $\mathbb{H}$, with the y-element vector, $[1, 0, ..., 0]$, where $y = max\{\boldsymbol{\xi}\}$. The most critical CCG is the one producing the least deviation from the reliability of the system with all CCGs active. Since simulation provides the time-dependent system reliability, the proposed approach can reveal the evolution, over time, of the relative criticality of the CCGs. The sensitivity of this relative criticality to the mean-time-to-failure (MTTF) of the component groups can also be investigated.

7.4 Case Studies

To illustrate how the proposed modelling and simulation approach is applied in practice, two case studies will be considered in this section. Though only numerical examples, the case studies have been carefully designed to reflect the every-day problems encountered by the system engineer in industry. I, therefore, believe they set the tone for the applicability of the proposed approach to realistic problems. To validate the approach, the solutions obtained are compared to the existing analytical survival signature-based approach [30,49], as well as a simulation approach based on a modification of the load-flow approach proposed in Chapters 3 and 4. The modified load-flow simulator is the same as Algorithm 11, save for the replacement of $\boldsymbol{S}_\tau(\underline{\boldsymbol{x}}')$ with a structure function that is assigned the value 1 for non-zero flows across the system and 0, otherwise.

Table 7.1: Failure time distribution data and CCF parameters of component groups.

Component Type	Distribution Type	Distribution Parameters	CCF Parameters
1	Weibull	(1.8,2.2)	{0.95, 0.05}
2	Exponential	1.2	{0.8, 0.1, 0.05, 0.05}
3	Weibull	(2.3,1.6)	{1}
4	Weibull	(3.2,2.6)	{0.9, 0.1}
5	Exponential	2.1	{0.75, 0.1, 0.1, 0.05}

7.4.1 Case Study 1: A Complex Bridge Network

Shown in Figure 7.1 is an arbitrary 13-component complex system, which components are arranged into five groups. The number within each box denotes which group the component belongs to while the number outside defines the index of the component in the system. Components of the same group are assumed to have the same failure time distribution, as defined in Table 7.1. The CCF parameters define the probabilities of a given number of components being affected by a CCF event and correspond to the α-factor CCF model. In the table, an exponential distribution is defined by its mean (in hours) while a Weibull distribution is defined by a set in which the first element is its scale parameter (in hours) and the second element, its shape parameter.

7.4.1.1 Analyses and Results

The system is first analysed with and without CCF, using the proposed simulation model and the data in Table 7.1. For this system, $\boldsymbol{\xi} = \{2, 4, 1, 2, 4\}$ and the CCF matrices, with and without CCF, are as expressed in Equations 7.3 and 7.4, respectively.

$$\mathbb{H} = \begin{pmatrix} 0.95 & 0.05 & 0 & 0 \\ 0.8 & 0.1 & 0.05 & 0.05 \\ 1 & 0 & 0 & 0 \\ 0.9 & 0.1 & 0 & 0 \\ 0.75 & 0.1 & 0.1 & 0.05 \end{pmatrix} \tag{7.3}$$

$$\mathbb{H} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \tag{7.4}$$

To validate the proposed approach and demonstrate its generality, it was used to analyse the system, with and without dependencies. The system was then re-analysed, in separate instances, using load-flow simulation and an analytical algorithm based on Equation 2.3 in Section 2.5.1. The analytical algorithm was used only for the case

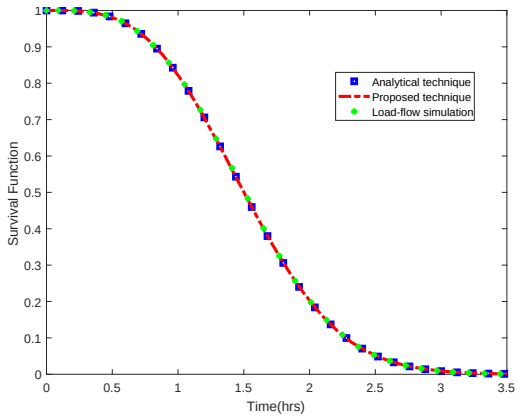


Figure 7.2: System reliability with dependencies ignored.

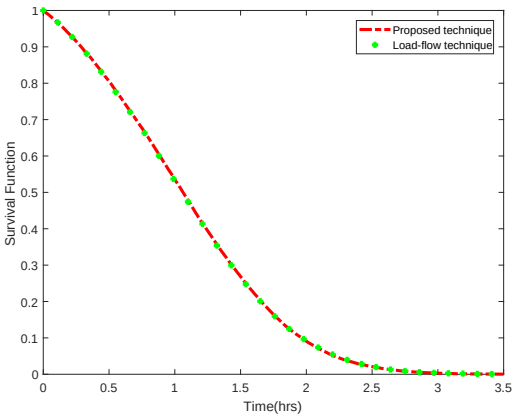


Figure 7.3: System reliability with dependencies considered.

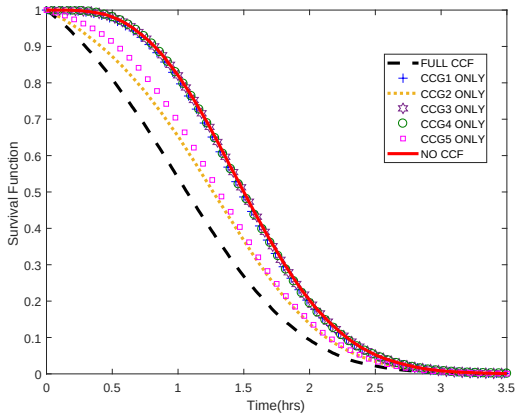


Figure 7.4: System reliability sensitivity to CCGs.

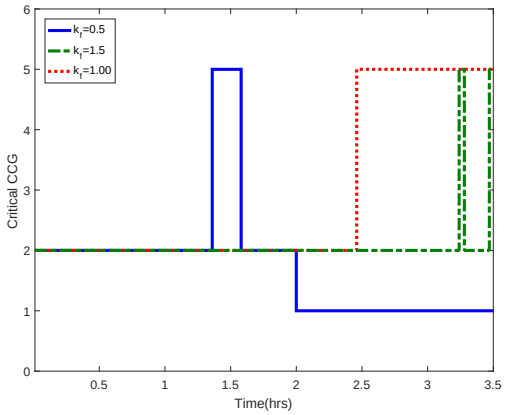


Figure 7.5: Sensitivity of critical CCG to component MTTF.

without dependencies, due to its inapplicability to the other. Figures 7.2 and 7.3 show the system reliability plots for a mission time of 3.5 hours and 5×10^4 samples.

The relative sensitivity of the system survival function to the Common-Cause Groups (CCG) was also investigated. The system was analysed considering CCF in all the CCGs (designated full CCG), with no CCF, and CCF in only one group at a time, for all the CCGs. The results obtained are plotted in Figure 7.4, from which the critical CCG is deduced, as described in Section 7.3.5. Figure 7.5 shows the variation of this CCG with component mean-time-to-failure (MTTF), as a function of time. The factor, k_f, denotes the number by which the nominal MTTF presented in Table 7.1 is multiplied. For an exponential distribution, the new mean becomes $\lambda_0 k_f$, where λ_0 is its nominal mean. The MTTF of a component with a failure time following a Weibull distribution is varied by keeping the shape parameter constant, while varying the scale parameter. If α_0 is the nominal scale parameter, the new scale parameter becomes $\alpha_0 k_f$.

Table 7.2: Comparison of computation times (in seconds).

	Dependencies Ignored	With Dependencies
Survival Signature computation	92.00	92.00
Analytical technique	0.29	n/a
Proposed technique	32.50	32.80
Load-Flow simulation	2300.00	1654.00

7.4.1.2 Discussions

The accuracy and generality of the proposed simulation approach are validated by the plots in Figures 7.2 and 7.3, given the agreement between the results yielded by the various techniques. As highlighted in Figure 7.4, the reliability of the system reduces drastically when the effects of CCF are factored into the analysis. It exemplifies the need to consider this realistic aspect of a system's operation in its reliability evaluation. The figure also reveals CCG-2 as the most critical and CCG-3, the least critical. The latter, however, is not surprising, as CCG-3 is made up of only one component, inferring its immunity to CCF. Figure 7.5 shows that the criticality of a CCG may or may not remain fixed during the mission, depending on the MTTF of its components. For instance, for $k_f = 1$, corresponding to the nominal values presented in Table 7.1, CCG-2 is initially the most critical until at time, $t = 2.5$ hours, when it is overtaken by CCG-5. A different trend, however, is realised with $k_f = 0.5$, for instance. The essence of the results presented in Figure 7.5 could be appreciated on two fronts;

1. Since the most critical CCG is a function of the MTTF of its components, there is sufficient incentive for the system operator to re-identify the most critical CCG after every component replacement.

2. Given limited resources, the operator can efficiently allocate mitigating resources during the mission. For instance, with $k_f = 0.5$, resources should be allocated to CCG-2 for $0 \leqslant t \leqslant 1.36$, CCG-5 for $1.36 \leqslant t \leqslant 1.57$, and CCG-1 for $t \geqslant 2$.

Though the proposed approach yields the same outcome as the load-flow simulation and analytical techniques, it requires less computational effort than the former but more than the latter, as summarised in Table 7.2. The table provides the recorded wall clock times for each approach, when the system was analysed on a 2GHz Intel(R) Core(TM) i5-4590T computer. Row 1 of the table provides the time it took to derive the survival signature of the system, for all its possible state vectors. This time is fixed, with or without dependencies, since the survival signature depends only on the system topology.

In survival signature-based techniques, the structure function of a system is computed once for each of its state vectors. In load-flow simulation, however, the simulation computes the flow through the system for every component failure. Therefore, there could be up to N load-flow calculations per state vector, in an N sample simulation. This explains why the proposed approach is more efficient, even when both

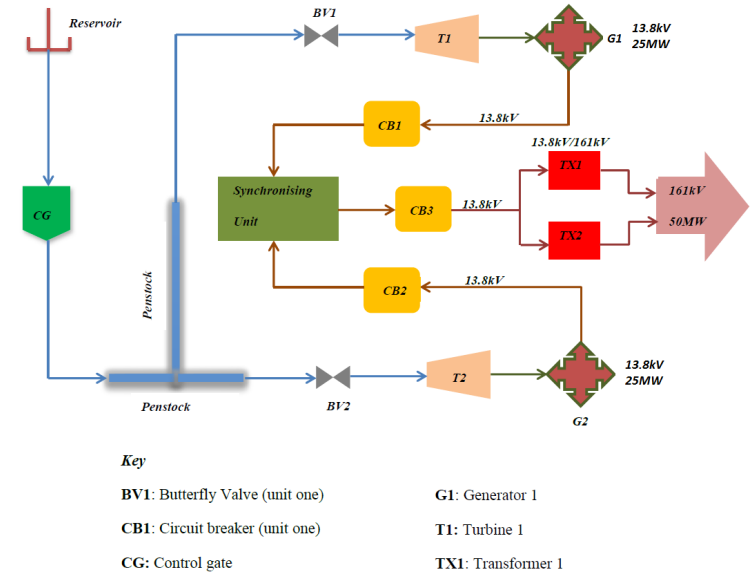


Figure 7.6: Schematic of a 50MW hydroelectric power plant.

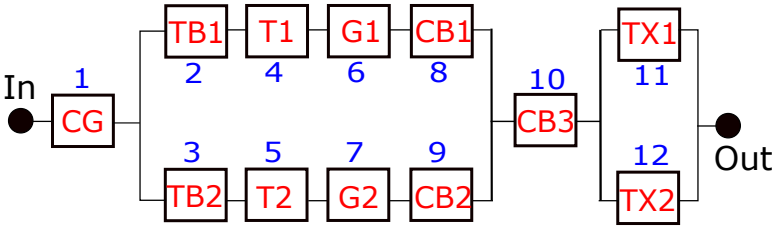


Figure 7.7: Condensed block diagram of the plant.

are simulation-based. Its computational superiority is most appreciated when employed to solve a problem requiring repeated system evaluations. For instance, if the system under consideration were analysed n times with dependencies ignored, the proposed approach would take $92 + 32.5n$ seconds and the load-flow simulation, $2300n$ seconds.

7.4.2 Case Study 2: A Hydroelectric Power Plant

In this case-study, a two-unit hydroelectric power plant adapted from the first case study in Chapter 5, and which schematic is shown in Figure 7.6, is analysed. It is a slightly modified model of the Bumbuna hydroelectric power plant; a 50MW plant in Sierra Leone. Its two units are similar, and each, rated 25MW consists a butterfly valve, turbine, generator, and circuit breaker. The power generated by the units is synchronized in the synchronizing unit and fed to the step-up transformers for onward transmission. The equivalent reliability block diagram of the plant is presented in Figure 7.7, where the Penstock and Synchronising unit have been neglected due to their very high reliability. CB3, which has two failure modes, is of a make different from that of CB1 and CB2. Its failure in mode 1 forces the failure of CB1 while its failure in

Table 7.3: Failure time distribution data and CCF parameters of plant components.

Component Type	Components	Distribution Type	Distribution Parameters	CCF Parameters
1	1	Weibull	$(3, 1.8)$	$\{1, 0\}$
2	2,3	Weibull	$(1.8, 2.3)$	$\{0.9, 0.1\}$
3	4,5	Weibull	$(4, 3)$	$\{0.8, 0.2\}$
4	6,7	Weibull	$(2.1, 2.6)$	$\{0.85, 0.15\}$
5	8,9	Exponential	4	$\{0.8, 0.2\}$
6	10	Exponential	3.85	$\{1, 0\}$
7	11,12	Gamma	$(3, 1)$	$\{0.82, 0.18\}$
8	13	Hazard Function	$2t$	$\{1, 0\}$
9	14	Hazard Function	$t^2 + t/100$	$\{1, 0\}$

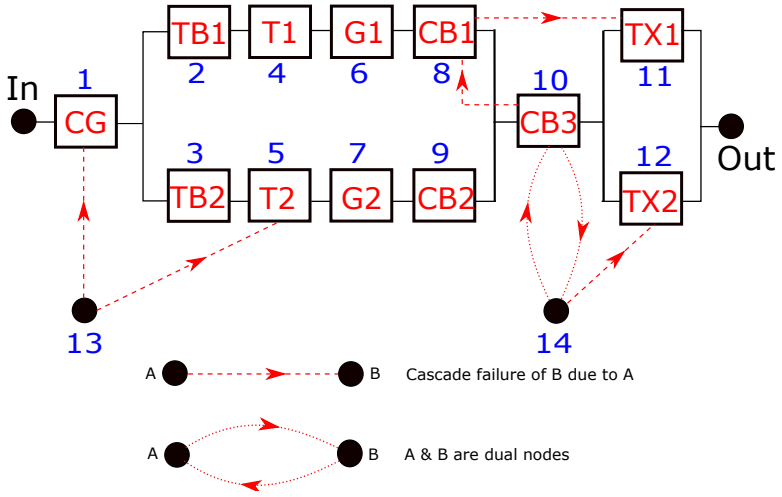


Figure 7.8: Plant block diagram showing interdependencies.

mode 2 forces the failure of TX2. The dam is contaminated with impurities that induce failure in CG and T2, at a rate of $2t$ per year, where t is the time spent in operation. T1, however, is conservatively assumed immune to this impurity. The goal of this case study is to compute the reliability of the plant for a mission time of two years.

$$\begin{aligned} f_k(t) &= h_k(t)\, e^{-\int_0^t h_k(u)du} \\ F_k(t) &= 1 - e^{-\int_0^t h_k(u)du} \end{aligned} \tag{7.5}$$

$$\mathbb{H} = \begin{pmatrix} 1 & 0 \\ 0.9 & 0.1 \\ 0.8 & 02 \\ 0.85 & 0.15 \\ 0.8 & 0.2 \\ 1 & 0 \\ 0.82 & 0.18 \\ 1 & 0 \\ 1 & 0 \end{pmatrix} \qquad \mathbb{C} = \begin{matrix} (8, 11) & 1 \\ (10, 8) & 1 \\ (13, 1) & 1 \\ (13, 5) & 1 \\ (14, 12) & 1 \end{matrix} \tag{7.6}$$

Table 7.4: Comparison of computation times (in seconds).

	Dependencies Ignored	With Dependencies
Survival Signature	34.20	34.20
Analytical technique	0.07	n/a
Proposed technique	29.00	32.80
Load-Flow simulation	757.70	486.40

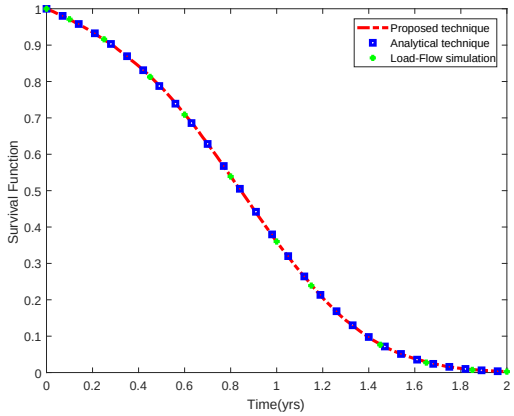


Figure 7.9: Plant reliability with dependencies ignored.

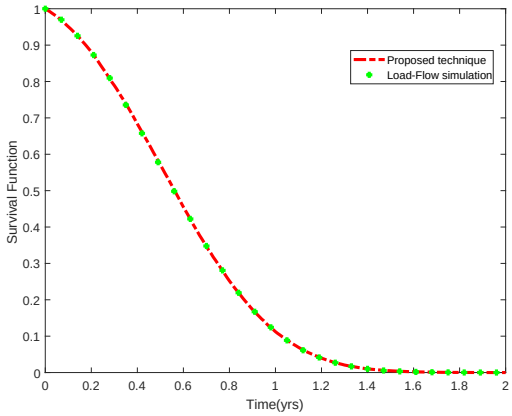


Figure 7.10: Plant reliability with dependencies considered.

7.4.2.1 Analyses and Results

The impurity affecting CG & T2 and the second failure mode of CB3 can each be represented by an external node, as proposed in Section 7.3.1. The impurity is assigned a component ID of 13 and failure mode 2 of CB3, a component ID of 14. The latter is also the dual of node 10, since they both represent different failure modes of the same component, CB3. In Figure 7.8 is the final block diagram of the plant, showing cascading dependencies. The components, including the external nodes have been organised into 9 component groups/types depending on their similarity in functionality, make, and failure characteristics. Table 7.3 presents these component groups, their composition, failure time distribution (in years), and CCF parameters. In the table, "Hazard Function" as a distribution type suggests only the hazard rate, $h_k(t)$, of failures is known for that component type. Given $h_k(t)$, however, the probability density function, $f_k(t)$, and the cumulative density function, $F_k(t)$, for type k can be obtained as in Equation 7.5. For the plant, $\boldsymbol{\xi} = \{1, 2, 2, 2, 2, 2, 2, 1, 1\}$, $\mathbb{D}_{10} = \{14\}$, $\mathbb{D}_{14} = \{10\}$, and $\mathbb{D}_i = \varnothing \forall i \notin \{10, 14\}$. The plant's CCF matrix, $\mathbb{H}$ and cascade matrix, $\mathbb{C}$, defined as a sparse matrix, are given in Equation 7.6.

As in the first case study, the plant was analysed using 5×10^4 simulation samples and the same reliability evaluation techniques used in that case study, with dependencies neglected. It was then re-analysed using the proposed technique and load-flow simulation, but this time considering dependencies. The computation time in each case, in seconds of wall clock time, was recorded as presented in Table 7.4 while Figures 7.9

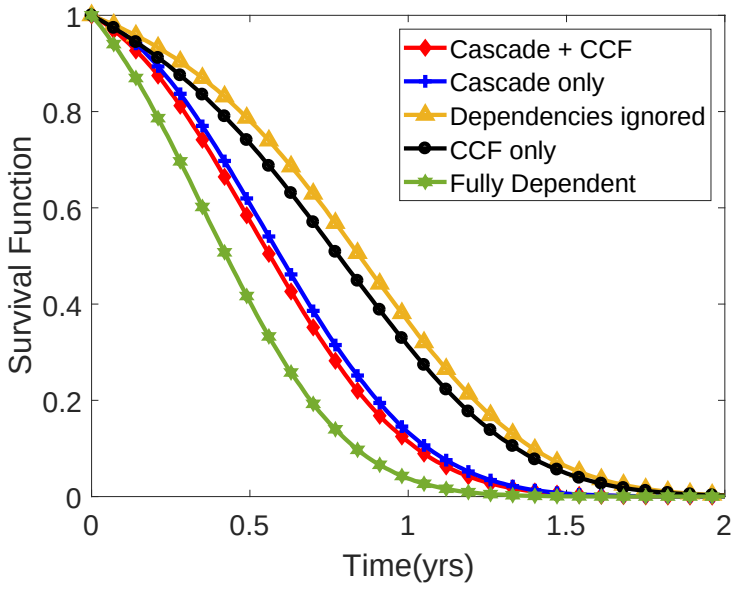


Figure 7.11: The effects of dependencies on plant reliability.

and 7.10 show the plots of the results obtained. The plant was also analysed separately, with CCF neglected (denoted "Cascade only") and then with cascading failures neglected (denoted "CCF only"). The results obtained were plotted on the same axes, as shown in Figure 7.11, to lay bare, the effects of dependencies, as well as the relative influence of CCF and cascading failures on the reliability of the plant. The plot designated 'Fully Dependent' denotes the reliability of the plant when the failure of any component triggers the simultaneous failure of all the remaining components of the plant. The assumption, in other words, effectively reduces the plant to a series-connected system.

7.4.2.2 Discussions

Figures 7.9 and 7.10 further validate the accuracy and generality of the proposed approach. As in case study 1, Table 7.4 shows the proposed approach to be somewhere between the analytical and load-flow techniques, in terms of computational efficiency. As expected, the reliability of the plant is maximum, with dependencies neglected and least, otherwise. The effects of cascading failures, however, are prominent, according to Figure 7.11. This can be attributed to the effects of the contamination of the dam on the control gate (CG), since the failure of the latter induces the failure of the plant. The reliability of the plant when it is assumed independent and that when it is assumed fully dependent, represent the bounds on its reliability. These bounds can ensure an informed decision when the full extent and nature of interdependencies are unknown.

7.5 Chapter Summary

Dependent failures can impose adverse effects on the reliability and performance of a multi-component system. They are normally a consequence of functional and induced couplings between components, due to a variety of possible reasons. Thus, there is an inevitability about the susceptibility of realistic multi-component engineering systems to them. The need, therefore, to incorporate dependency considerations into system reliability analysis is overwhelming, if an accurate reliability estimate is required.

In this chapter, an approach that extends the applicability of the system survival signature-based approach to system reliability evaluation to systems susceptible to dependent failures has been proposed. Being a hybrid technique, it inherits the desirable attributes of both the system survival signature and Monte Carlo Simulation. Consequently, it overcomes the issues of topological complexity, diversity in component failure time distributions, and complexities in inter-component interactions. Since the survival signature of a system is computed prior to its reliability evaluation and given this signature is static for a fixed system topology, the proposed approach is ideal for reliability/maintenance optimization and sensitivity/uncertainty analyses. The approach has been shown to be computationally efficient, albeit less efficient than the analytical approach, which, however, is inapplicable to systems with dependent failures. This leaves the proposed approach the most efficient alternative for realistic engineering systems.

It is probably noteworthy to clarify at this point that the Common-Cause and cascading failure modelling strategies proposed in Chapters 4 and 6 could have been used to model dependencies in the proposed approach. However, those modelling strategies are optimized for multi-state components, and are, therefore, a little complicated for binary-state components. Since survival signature is a terminology suited to binary-state systems, it was necessary to develop their simpler binary-state equivalents.

This chapter has also shown how the approach is used to obtain key system reliability indices. Knowledge of the most critical Common-Cause Group, for instance, and how it varies with time and component MTTF, could influence the limited resource allocation in the mitigation of Common-Cause Failures. The proposed approach, therefore, can be used as a decision-support tool in the operation and management of complex systems.

Chapter 8

Conclusions

8.1 Concluding Remarks

Public resistance to nuclear power generation has increased dramatically, since the Fukushima Daiichi incident in 2011 [67]. Communities are gradually becoming wary of its evident risks, which wariness is exacerbated by the growing prospect of nuclear proliferation by rogue states. Consequently, states have become genuinely cautious about investing in new nuclear power technologies. Taiwan, for instance, halted the construction of its fourth nuclear power plant in 2014, following a public outcry triggered by the Fukushima incident. Engineers, therefore, are challenged with the responsibility of improving the safety of nuclear power plants and assuring the public of minimal consequences, should something go wrong. This calls for the design of reliable and resilient systems, which are subjected to a constant performance monitoring and evaluation.

Probabilistic Risk Assessment (PRA) is a framework that assesses what can go wrong in a system with its ensuing consequences, and is in extensive use in the nuclear power industry. It has been used for regulatory purposes, among other tasks, to compare designs and to obtain relevant licenses. System reliability analysis, which investigates how well a system performs its intended functions, sits at the heart of this framework. The quality and computational cost, therefore, of a PRA, is influenced by the robustness, as well as the intuitiveness of its underlying system reliability modelling and evaluation scheme. When modelling the reliability of a system, the scheme used would depend largely on the complexity of the system, the level of accuracy demanded, the reliability indices required, and the preference of the analyst. Since every scheme has its own merits and demerits, selecting the best, therefore, requires a lot of trade-off considerations.

System complexity, in my opinion, is the greatest determinant of which scheme is most plausible for the reliability analysis of a given system. Complexity here encompasses topological complexities, component interdependencies, multi-state component attributes, complex maintenance strategies, and any other attribute which inhibits the application of simple reliability engineering reasoning to the system. For these systems,

research places simulation-based schemes ahead, for a realistic analysis of their reliability, despite the relatively higher computational intensity of simulation-based schemes. This, however, is not a problem, given the emergence of parallel computing technology.

The applicability of the existing simulation frameworks, however, is inhibited when the system being analysed can exist in multiple performance levels. In this case, all the performance levels would have to be known/deduced in advance and a multi-state fault tree defined for each. This is not only a painstaking process, but one with the capacity to inadvertently introduce errors in the reliability evaluation process. In addition, the reliability of systems prone to flow losses and systems that implement reconfiguration following certain failure events, can not be easily computed by these frameworks.

In this thesis, therefore, an intuitive simulation framework, applicable to binary and multi-state systems of any topology, has been developed. It does not require the prior definition of the structure function, minimal cut sets, or the minimal path sets of the system. Instead, it employs a linear programming algorithm and the principles of flow conservation to compute the actual flow through the system. Thus, it can model flow losses and implement reconfiguration requirements relatively easily. It also accounts for all forms of interdependencies in realistic systems using an intuitive matrix representation. These attributes render the framework intuitive and generally applicable.

Maintenance being a key determinant of the reliability and resilience of an engineering system, the framework is equipped with a robust maintenance modelling and optimization scheme. The scheme takes cognisance of realistic operational constraints like limited maintenance teams, delayed availability of spares, priority maintenance, and operational uncertainties. The climax, arguably, of this doctoral research is the application of the simulation framework developed, to model station blackout accidents in nuclear power plants. Illustrated, also in this thesis, is how the results obtained can be incorporated into the existing PRA framework, highlighting the wide applicability, as well as compatibility with legacy techniques, of the simulation framework.

While the framework is optimized for multi-state systems, it can be grossly inefficient for binary-state systems in which at least 2 components are identical. For this reason, a survival signature-based simulation approach was developed in conjunction, for non-repairable systems susceptible to common-cause and cascading failures. The survival signature of a system is a function of its topology only, and, therefore, unique to that system. Consequently, it is calculated just once and reused in multiple reliability analyses of the same system. This feature effectively reduces the reliability evaluation of the system to the analysis of the failure probabilities of its component, which is computationally much cheaper. Though applied to non-repairable systems in this doctoral study, the approach is extendible to maintainable systems without loss of generality.

In summary, the proposed framework can be used to model and evaluate the reliability of complex systems, identify complex maintenance strategies, aid in the level 1 probabilistic risk assessment of nuclear power plants, and even mitigate the impact of

targeted attacks on systems. The framework is implemented in the open-source uncertainty quantification toolbox, OpenCossan [109,110], to render it readily available.

8.2 Recommendations for Future Work

This doctoral study was aimed at developing a robust and reliable computational framework that transcends the existing frameworks for the reliability analysis of realistic engineering systems. Though this, as highlighted in the preceding section, has been achieved, there still is room for improvement. The following, therefore, is an overview of the relevant shortfalls of the framework developed, with recommendations.

1. Like all simulation-based techniques, the framework developed is computationally demanding. It is, therefore, prudent to explore tangible avenues of improving its computational efficiency. One of these could be:

   (a) developing an analytical approximation to the framework;

   (b) employing machine learning or other techniques with the potential to reduce the number of load-flow calculations per sample or the total number of samples required for an accurate outcome;

   (c) developing an analytical survival signature-based approach to model systems susceptible to both common-cause and cascading failures.

2. The maintenance strategy optimization scheme proposed identifies the best strategy from a predefined set of strategies. This implies, the optimal strategy yielded is only the local optima, which can also be the true optima but only if the predefined set contains the optimal strategy. I am of the opinion that a single maintenance strategy cannot yield the absolute optimal system performance. Instead, a set of strategies is required, such that certain strategies are useful only within certain periods of the mission. For instance, only a few maintenance resources are required at the beginning of a mission, when the components are still new. As the mission progresses, however, the components start ageing and the maintenance resources required for optimal system performance increase. It suffices to say, therefore, that the optimal number of maintenance teams is time-dependent. Consequently, assigning a fixed number of maintenance teams to the system may result in the incurring of costs that would otherwise be prevented by a dynamic allocation, especially at the early stages of the mission. A real-time maintenance optimization scheme, where maintenance decisions are made during system analysis, therefore, is recommended. The maintenance decisions should take cognisance of the current state of the system, the remaining lives of its components, and the outcomes of similar decisions made in the past. This requirement, however, may necessitate the invocation of artificial intelligence techniques. The scheme, also,

would require just one system evaluation, compared to the tens or even hundreds (depending on the system and the number of maintenance strategies) of system evaluations the current scheme employs.

3. Though not illustrated in this thesis, the developed framework can also be used to evaluate the reliability of a system with epistemic uncertainties in the failure parameters of its components. This is achieved via repeated model evaluations, in a double loop Monte Carlo scheme. The scheme is designated so because, the framework being repeatedly evaluated (inner loop) is Monte Carlo-based, and so is the outer loop sampling the parameters used to evaluate it. It is, therefore, not difficult to realise the inefficiency of this scheme and the need for a better one. Modelling the imprecise component failure parameters as p-boxes [50,65] or intervals [80] and propagating them through the framework may be computationally more efficient. The task, therefore, is finding the best way to go about this.

4. In most critical applications, the system operator is most interested in how quickly the system recovers after failure or how quickly it reacts to mitigate the consequences of an undesirable event. The ability of a system to maintain its normal operation following a perturbation is a measure of its resilience. The outcomes of the application of the framework developed in this doctoral study to a reconfigurable offshore oil installation and AC power recovery during station blackout accidents, have shown its potential applicability to resilience engineering. In both applications, however, the reconfiguration requirements and the sequence of operator response were predefined. This is not the case in practice, especially when the set of possible reconfiguration options or operator response sequences is large or unbounded. A real-time decision support tool, therefore, is recommended. Such a tool can provide the best system reconfiguration procedure and the optimal sequence of operator response, following an undesirable event. Since the exact status of the system is in most cases not known during a crisis, the tool should be capable of modelling the epistemic uncertainties in the current states of the system and its components and make a decision in the presence of these uncertainties.

Bibliography

[1] M. Abd-El-Barr. *Design and analysis of reliable and fault-tolerant computer systems.* Hackensack, NJ, USA: World Scientific Publishing Co., 2006.

[2] Takao Adachi and Bruce R. Ellingwood. Serviceability of earthquake-damaged water systems: Effects of electrical power availability and power backup systems on system vulnerability. *Reliability Engineering & System Safety*, 93(1):78 – 88, 2008.

[3] Ahmad W. Al-Dabbagh and Lixuan Lu. Reliability modeling of networked control systems using dynamic flowgraph methodology. *Reliability Engineering and System Safety*, 95(11):1202 – 1209, 2010.

[4] Wajdi Al-Khateeb, Khalid Al-Khateeb, and Sufyan Al-Irhayim. Availability evaluation of scalable complex networks. In *International Conference on Computer and Communication Engineering, ICCCE 2012*, pages 966–971, 2012.

[5] John Andrews and Claudia Fecarotti. System design and maintenance modelling for safety in extended life operation. *Reliability Engineering & System Safety*, 163:95 – 108, 2017.

[6] John Andrews, Darren Prescott, and Florian De Rozières. A stochastic model for railway track asset management. *Reliability Engineering & System Safety*, 130:76 – 84, 2014.

[7] Louis J.M. Aslett, Frank P.A. Coolen, and Simon P Wilson. Bayesian inference for reliability of systems and networks using the survival signature. *Risk Analysis*, 35(3):1640–1651, 2015.

[8] Siu-Kui Au and James L. Beck. Estimation of small failure probabilities in high dimensions by subset simulation. *Probabilistic Engineering Mechanics*, 16(4):263 – 277, 2001.

[9] J. Barata, C.Guedes Soares, M. Marseguerra, and E. Zio. Simulation modelling of repairable multi-component deteriorating systems for on-condition maintenance optimisation. *Reliability Engineering & System Safety*, 76(3):255 – 264, 2002.

[10] RE Barlow and F Proschan. *Mathematical Theory of Reliability*. New York: Wiley, 1965.

[11] R Billinton and N.A Ronald. *Reliability Evaluation of Engineering Systems*, pages 81–306. New York and London: Plenum Press, 1992.

[12] A. Bobbio, L. Portinale, M. Minichino, and E. Ciancamerla. Improving the analysis of dependable systems by mapping fault trees into Bayesian networks. *Reliability Engineering & System Safety*, 71(3):249 – 260, 2001.

[13] Ettore Bompard, Ciwei Gao, Roberto Napoli, Angela Russo, Marcelo Masera, and Alberto Stefanini. Risk assessment of malicious attacks against power systems. *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, 39(5):1074 – 1085, 2009.

[14] E. Borgonovo, M. Marseguerra, and E. Zio. A Monte Carlo methodological approach to plant availability modeling with maintenance, aging and obsolescence. *Reliability Engineering & System Safety*, 67(1):61 – 73, 2000.

[15] Marc Bouissou, Hilding Elmqvist, Martin Otter, and Albert Benveniste. Efficient monte carlo simulation of stochastic hybrid systems. In *Proceedings of the 10th International Modelica Conference; March 10-12; 2014; Lund; Sweden*, number 96, pages 715–725. Linkoping University Electronic Press; Linkopings Universitet, 2014.

[16] British Broadcasting Corporation. The Chernobyl disaster. `http://news.bbc.co.uk/1/shared/spl/hi/guides/456900/456957/html/nn1page1.stm`. Online; accessed 02-04-2018.

[17] British Broadcasting Corporation. Timeline: BP oil spill. `http://www.bbc.co.uk/news/world-us-canada-10656239`, September 2010. Online; accessed 29-03-2018.

[18] British Broadcasting Corporation. Counting the cost of the BP disaster one year on. `http://www.bbc.co.uk/news/business-13120605`, April 2011. Online; accessed 29-03-2018.

[19] British Broadcasting Corporation. Timeline: Nuclear plant accidents. `http://www.bbc.co.uk/news/world-13047267`, September 2011. Online; accessed 31-03-2018.

[20] British Broadcasting Corporation. Fukushima report: key points in nuclear disaster report. `http://www.bbc.co.uk/news/world-asia-18718486`, July 2012. Online; accessed 03-04-2018.

[21] British Broadcasting Corporation. British airways flight chaos lessens after weekend of disruption. `http://www.bbc.co.uk/news/uk-40081112`, May 2017. Online; accessed 29-03-2018.

[22] British Broadcasting Corporation. British airways owner IAG says IT chaos cost £58m. `http://www.bbc.co.uk/news/business-40750168`, July 2017. Online; accessed 29-03-2018.

[23] Sergey V. Buldyrev, Roni Parshani, Gerald Paul, H. Eugene Stanley, and Shlomo Havlin. Catastrophic cascade of failures in interdependent networks. *Nature*, 464(7291):1025–1028, April 2010.

[24] Sungil Byun, Inseok Yang, Moo Geun Song, and Dongik Lee. Reliability evaluation of steering system using dynamic fault tree. In *IEEE Intelligent Vehicles Symposium*, pages 1416 – 1420, 2013.

[25] Marko Čepin. *Assessment of Power System Reliability: Methods and Applications*, chapter Event Tree Analysis, pages 89–99. Springer London, London, 2011.

[26] Marko Čepin. *Assessment of Power System Reliability: Methods and Applications*, chapter Fault Tree Analysis, pages 61–87. Springer London, London, 2011.

[27] Marko Čepin. *Assessment of Power System Reliability: Methods and Applications*, chapter Distribution and Transmission System Reliability Measures, pages 215–226. Springer, London, 2011.

[28] Marko Čepin and Borut Mavko. A dynamic fault tree. *Reliability Engineering & System Safety*, 75(1):83 – 91, 2002.

[29] Zheng Chen and Toby Berger. Reliability and availability analysis of manhattan street networks. *IEEE Transactions on Communications*, 42(-4):511–522, 1994.

[30] Frank P. A. Coolen and Tahani Coolen-Maturi. Generalizing the signature to systems with multiple types of components. In Wojciech Zamojski, Jacek Mazurkiewicz, Jarosław Sugier, Tomasz Walkowiak, and Janusz Kacprzyk, editors, *Complex Systems and Dependability*, pages 115–130, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[31] Frank P.A. Coolen and Tahani Coolen-Maturi. Predictive inference for system reliability after common-cause component failures. *Reliability Engineering & System Safety*, 135:27 – 33, 2015.

[32] Marco de Angelis, Edoardo Patelli, and Michael Beer. Advanced line sampling for efficient robust reliability analysis. *Structural Safety*, 52:170 – 182, 2015.

[33] Rommert Dekker. Applications of maintenance optimization models: a review and analysis. *Reliability Engineering & System Safety*, 51(3):229 – 240, 1996. Maintenance and reliability.

[34] William Denson. History of reliability prediction. *IEEE Transactions on Reliability*, 47(3-SP):321 – 328, 1998.

[35] F.A. Van der Duyn Schouten and S.G. Vanneste. Maintenance optimization of a production system with buffer capacity. *European Journal of Operational Research*, 82(2):323 – 338, 1995. OR Models for Maintenance Management and Quality Control.

[36] BS Dhillon and OC Anude. Common-cause failures in engineering systems: A review. *International Journal of Reliability, Quality and Safety Engineering*, 1(01):103–129, 1994.

[37] S.V. Dhople, L. DeVille, and A.D. Dominguez-Garcia. A stochastic hybrid systems framework for analysis of markov reward models. *Reliability Engineering & System Safety*, 123:158 – 170, 2014.

[38] Salvatore Distefano and Antonio Puliafito. Reliability and availability analysis of dependent-dynamic systems with DRBDs. *Reliability Engineering and System Safety*, 94(9):1381–1393, 2009.

[39] J. B. Dugan, S. J. Bavuso, and M. A. Boyd. Dynamic fault-tree models for fault-tolerant computer systems. *IEEE Transactions on Reliability*, 41(3):363–377, Sep 1992.

[40] J. B. Dugan, K. J. Sullivan, and D. Coppit. Developing a low-cost high-quality software tool for dynamic fault-tree analysis. *IEEE Transactions on Reliability*, 49(1):49–59, Mar 2000.

[41] Joanne Bechta Dugan, Salvatore J. Bavuso, and Mark A. Boyd. Fault trees and markov models for reliability analysis of fault-tolerant digital systems. *Reliability Engineering & System Safety*, 39(3):291 – 307, 1993.

[42] Ajendra Dwivedi and Xinghuo Yu. A maximum-flow-based complex network approach for power system vulnerability analysis. *IEEE Transactions on Industrial Informatics*, 9(1):81 – 88, 2013.

[43] S A Eide, C D Gentillon, T E Wierman, and D M Rasmuson. Reevaluation of station blackout risk at nuclear power plants. Technical Report NUREG/CR-6890 Vol. 2, U.S. Nuclear Regulatory Commission, 2005.

[44] S A Eide, C D Gentillon, T E Wierman, and D M Rasmuson. Reevaluation of station blackout risk at nuclear power plants. Technical Report NUREG/CR-6890 Vol. 1, U.S. Nuclear Regulatory Commission, 2005.

[45] Serkan Eryilmaz. The concept of weak exchangeability and its applications. *Metrika*, 80(3):259–271, Apr 2017.

[46] Serkan Eryilmaz, Frank P.A. Coolen, and Tahani Coolen-Maturi. Marginal and joint reliability importance based on survival signature. *Reliability Engineering & System Safety*, 172:118 – 128, 2018.

[47] Serkan Eryilmaz, Frank P.A. Coolen, and Tahani Coolen-Maturi. Mean residual life of coherent systems consisting of multiple types of dependent components. *Naval Research Logistics (NRL)*, 65(1):86–97, 2018.

[48] Mengfei Fan, Zhiguo Zeng, Enrico Zio, Rui Kang, and Ying Chen. A stochastic hybrid systems model of common-cause failures of degrading components. *Reliability Engineering & System Safety*, 172:159 – 170, 2018.

[49] Geng Feng, Edoardo Patelli, Michael Beer, and Frank P.A. Coolen. Imprecise system reliability and component importance based on survival signature. *Reliability Engineering & System Safety*, 150:116 – 125, 2016.

[50] Scott Ferson, Vladik Kreinovich, Lev Ginzburg, Davis S. Myers, and Kari Sentz. Constructing probability boxes and dempster-shafer structures. Technical report, Sandia National Laboratories, 2003.

[51] Ilia Frenkel, Anatoly Lisnianski, and Lev Khvatskin. On the lz-transform application for availability assessment of an aging multi-state water cooling system for medical equipment. In *Applied Reliability Engineering and Risk Analysis: Probabilistic Models and Statistical Inference*, chapter 5, pages 59–77. Wiley-Blackwell, 2013.

[52] H. George-Williams and E. Patelli. Maintenance strategy optimization for complex power systems susceptible to maintenance delays and operational dynamics. *IEEE Transactions on Reliability*, 66(4):1309–1330, Dec 2017.

[53] Hindolo George-Williams, Min Lee, and Edoardo Patelli. A framework for power recovery probability quantification in nuclear power plant station blackout sequences. In *Proceedings of the Probabilistic Safety Assessment and Management Conference*, volume 13, 2016.

[54] Hindolo George-Williams, Min Lee, and Edoardo Patelli. Probabilistic risk assessment of station blackout accidents in nuclear power plants. *IEEE Transactions on Reliability*, 67(2):494–512, 2018.

[55] Hindolo George-Williams and Edoardo Patelli. A hybrid load flow and event driven simulation approach to multi-state system reliability evaluation. *Reliability Engineering & System Safety*, 152:351 – 367, 2016.

[56] Hindolo George-Williams and Edoardo Patelli. Efficient availability assessment of reconfigurable multi-state systems with interdependencies. *Reliability Engineering and System Safety*, 15:431–444, 2017.

[57] Ahmad Ghaderi, M.R. Haghifam, and Seyed Mostafa Abedi. Application of Monte Carlo simulation in markov process for reliability analysis. In *IEEE 11th International Conference on Probabilistic Methods Applied to Power Systems, PMAPS 2010*, pages 293 – 298, Singapore, Singapore, 2010.

[58] J.H. Heo, M.K. Kim, and J.K. Lyu. Implementation of reliability-centered maintenance for transmission components using particle swarm optimization. *International Journal of Electrical Power & Energy Systems*, 55:238 – 245, 2014.

[59] Wang Hongzhou and Pham Hoang. Maintenance policies and analysis. In *Reliability and Optimal Maintenance*, Springer Series in Reliability Engineering, pages 31–49. Springer London, 2006.

[60] Chin Yu Huang and Yung Ruei Chang. An improved decomposition scheme for assessing the reliability of embedded systems by using dynamic fault trees. *Reliability Engineering & System Safety*, 92(10):1403 – 1412, 2007.

[61] Arne B. Huseby and Bent Natvig. Discrete event simulation methods applied to advanced importance measures of repairable components in multistate network flow systems. *Reliability Engineering & System Safety*, 119:186 – 198, 2013.

[62] International Atomic Energy Agency. Assessment of safety. `https://www.iaea.org/ns/tutorials/regcontrol/assess/assess3233.htm`. Online; accessed 02-04-2018.

[63] Jonas Johansson and Henrik Hassel. An approach for modelling interdependent infrastructures in the context of vulnerability analysis. *Reliability Engineering & System Safety*, 95(12):1335 – 1344, 2010. 19th European Safety and Reliability Conference.

[64] Bernhard Kaiser, Catharina Gramlich, and Marc Förster. State/event fault trees: A safety analysis model for software-controlled systems. *Reliability Engineering & System Safety*, 92(11):1521 – 1537, 2007.

[65] Durga Rao Karanki, Hari Shankar Kushwaha, Ajit Kumar Verma, and Srividya Ajit. Uncertainty analysis based on probability bounds (p-box) approach in probabilistic safety assessment. *Risk Analysis*, 29(5):662–675, 2009.

[66] Faisal I. Khan and S.A. Abbasi. Analytical simulation and PROFAT II: a new methodology and a computer automated tool for fault tree analysis in chemical process industries. *Journal of Hazardous Materials*, 75(1):1 – 27, 2000.

[67] Younghwan Kim, Minki Kim, and Wonjoon Kim. Effect of the Fukushima nuclear disaster on global public acceptance of nuclear energy. *Energy Policy*, 61:822 – 828, 2013.

[68] Masakazu Kojima, Shinji Mizuno, and Akiko Yoshise. A primal-dual interior point algorithm for linear programming. In Nimrod Megiddo, editor, *Progress in Mathematical Programming*, pages 29–47. Springer New York, 1989.

[69] W. Kuo and X. Zhu. Some recent advances on importance measures in reliability. *IEEE Transactions on Reliability*, 61(2):344–360, June 2012.

[70] Helge Langseth and Luigi Portinale. Bayesian networks in reliability. *Reliability Engineering & System Safety*, 92(1):92 – 108, 2007.

[71] K.E. Lansey, C. Basnet, L. W. Mays, and J. Woodburn. Optimal maintenance scheduling for water distribution systems. *Civil Engineering Systems*, 9(3):211–226, 1992.

[72] H. Lei and C. Singh. Non-sequential monte carlo simulation for cyber-induced dependent failures in composite power system reliability evaluation. *IEEE Transactions on Power Systems*, 32(2):1064–1072, March 2017.

[73] G. Levitin. Reliability evaluation for acyclic transmission networks of multi-state elements with delays. *Reliability, IEEE Transactions on*, 52(2):231–237, June 2003.

[74] G. Levitin and A. Lisnianski. Optimization of imperfect preventive maintenance for multi-state systems. *Reliability Engineering & System Safety*, 67(2):193 – 203, 2000.

[75] G. Levitin and A. Lisnianski. Multi-state system reliability analysis and optimization. In *Handbook of Reliability Engineering*, chapter 4, pages 61–90. Springer London, London, 2003.

[76] Gregory Levitin. A universal generating function approach for the analysis of multi-state systems with dependent elements. *Reliability Engineering & System Safety*, 84(3):285 – 292, 2004.

[77] Gregory Levitin. *The Universal Generating Function in Reliability Analysis and Optimization*. Springer-Verlag London Limited, 2005.

[78] Gregory Levitin, Anatoly Lisnianski, Hanoch Ben-Haim, and David Elmakis. Redundancy optimization for series-parallel multi-state systems. *IEEE Transactions on Reliability*, 47(2):165 – 172, 1998.

[79] Gregory Levitin, Liudong Xing, Hanoch Ben-Haim, and Yuanshun Dai. Multi-state systems with selective propagated failures and imperfect individual and group protections. *Reliability Engineering & System Safety*, 96(12):1657–1666, 12 2011.

[80] C. Li, X. Chen, X. Yi, and J. Tao. Interval-valued reliability analysis of multi-state systems. *IEEE Transactions on Reliability*, 60(1):323–330, March 2011.

[81] Jing-An Li, Yue Wu, Kin Keung Lai, and Ke Liu. Reliability estimation and prediction of multi-state components and coherent systems. *Reliability Engineering & System Safety*, 88(1):93 – 98, 2005.

[82] Wenjian Li and H. Pham. An inspection-maintenance model for systems with multiple competing processes. *Reliability, IEEE Transactions on*, 54(2):318–327, June 2005.

[83] Yi-Kuei Lin. A simple algorithm for reliability evaluation of a stochastic-flow network with node failure. *Computers & Operations Research*, 28(13):1277 – 1285, 2001.

[84] Yi-Kuei Lin. Using minimal cuts to evaluate the system reliability of a stochastic-flow network with failures at nodes and arcs. *Reliability Engineering & System Safety*, 75(1):41 – 46, 2002.

[85] A. Lisnianski. Lz-transform for a discrete-state continuous-time markov process and its applications to multi-state system reliability. In *Recent Advances in System Reliability; Signatures, Multi-state Systems and Statistical Inference*. Springer, 2012.

[86] A. Lisnianski and Y. Ding. Inverse Lz-transform for a discrete-state continuous-time markov process and its application to multi-state system reliability. In *Applied Reliability Engineering and Risk Analysis*. Wiley, 2014.

[87] A. Lisnianski, G. Levitin, and H. Ben-Haim. Structure optimization of multi-state system with time redundancy. *Reliability Engineering and System Safety*, 67(2):103 – 112, 2000.

[88] Anatoly Lisnianski. Extended block diagram method for a multi-state system reliability assessment. *Reliability Engineering & System Safety*, 92(12):1601 – 1607, 2007. Special Issue on ESREL 2005.

[89] Anatoly Lisnianski, Ilia Frenkel, and Yi Ding. Multi-state systems in nature and in engineering. In *Multi-state System Reliability Analysis and Optimization for Engineers and Industrial Managers*, pages 1–28. Springer London, London, 2010.

[90] Anatoly Lisnianski, Ilia Frenkel, Lev Khvatskin, and Yi Ding. Maintenance contract assessment for aging systems. *Quality and Reliability Engineering International*, 24(5):519–531, 2008.

[91] Yu Liu and Hong-Zhong Huang. Optimal replacement policy for multi-state system under imperfect maintenance. *Reliability, IEEE Transactions on*, 59(3):483–495, Sept 2010.

[92] Zengkai Liu, Yonghong Liu, Baoping Cai, Xiaolei Li, and Xiaojie Tian. Application of petri nets to performance evaluation of subsea blowout preventer system. *{ISA} Transactions*, 54:240 – 249, 2015.

[93] Zhenzhou Lu, Shufang Song, Zhufeng Yue, and Jian Wang. Reliability sensitivity method by line sampling. *Structural Safety*, 30(6):517 – 532, 2008.

[94] Viliam Makis and Andrew K.S. Jardine. A note on optimal replacement policy under general repair. *European Journal of Operational Research*, 69(1):75 – 82, 1993.

[95] Manish Malhotra and Kishor S. Trivedi. Dependability modeling using Petri-nets. *IEEE Transactions on Reliability*, 44(3):428 – 440, 1995.

[96] Adolfo Crespo Marquez and Antonio Sanchez Heguedas. Models for maintenance optimization: a study for repairable systems and finite time periods. *Reliability Engineering & System Safety*, 75(3):367 – 377, 2002.

[97] M Marseguerra and E Zio. Optimizing maintenance and repair policies via a combination of genetic algorithms and Monte Carlo simulation. *Reliability Engineering & System Safety*, 68(1):69 – 83, 2000.

[98] Marzio Marseguerra, Enrico Zio, and Luca Podofillini. Condition-based maintenance optimization by means of genetic algorithms and monte carlo simulation. *Reliability Engineering & System Safety*, 77(2):151 – 165, 2002.

[99] John J. McCall. Maintenance policies for stochastically failing equipment: A survey. *Management Science*, 11(5):493–524, 1965.

[100] Sanjay Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2(4):575–601, 1992.

[101] L. Meshkat, J. B. Dugan, and J. D. Andrews. Dependability analysis of systems with on-demand and active failure modes, using dynamic fault trees. *IEEE Transactions on Reliability*, 51(2):240–251, Jun 2002.

[102] Henri Métivier. Chernobyl: Assessment of radiological and health impacts. Technical report, Nuclear Energy Agency and Organisation for Economic Co-operation and Development, 2002.

[103] A Mosleh, KN Fleming, GW Parry, HM Paula, DH Worledge, and Dl M Rasmuson. Procedures for treating common cause failures in safety and reliability studies: Volume 1, procedural framework and examples: Final report. Technical report, Pickard, Lowe and Garrick, Inc., Newport Beach, CA (USA), 1988.

[104] A Mosleh, D M Rasmuson, and F M Marshall. Guidelines on modeling Commom-Cause Failures in probabilistic risk assessment. Technical Report NUREG/CR-5485, U.S. Nuclear Regulatory Commission, 1998.

[105] Ali Mosleh. Common cause failures: an analysis methodology and examples. *Reliability Engineering & System Safety*, 34(3):249–292, 1991.

[106] Andrew O'Connor and Ali Mosleh. A general cause based methodology for analysis of common cause and dependent failures in system risk and reliability assessments. *Reliability Engineering & System Safety*, 145:341 – 350, 2016.

[107] Christos Ordoudis, Pierre Pinsona, Juan M. Morales, and Marco Zugno. An updated version of the IEEE RTS 24-bus system for electricity market and power system operation studies. Technical report, Technical University of Denmark, 2016.

[108] Min Ouyang. Review on modeling and simulation of interdependent critical infrastructure systems. *Reliability Engineering & System Safety*, 121:43 – 60, 2014.

[109] Edoardo Patelli. *Handbook of Uncertainty Quantification*, chapter COSSAN: A Multidisciplinary Software Suite for Uncertainty Quantification and Risk Management, pages 1–69. Springer International Publishing, 2017.

[110] Edoardo Patelli, Matteo Broggi, Marco De Angelis, and Michael Beer. Opencossan: An efficient open tool for dealing with epistemic and aleatory uncertainties. In *Vulnerability, Uncertainty, and Risk: Quantification, Mitigation, and Management - Proceedings of the 2nd International Conference on Vulnerability and Risk Analysis and Management, ICVRAM 2014 and the 6th International Symposium on Uncertainty Modeling and Analysis, ISUMA 2014*, pages 2564 – 2573, 2014.

[111] Edoardo Patelli, Geng Feng, Frank PA Coolen, and Tahani Coolen-Maturi. Simulation methods for system reliability using the survival signature. *Reliability Engineering & System Safety*, 167:327–337, 2017.

[112] Probability Methods Subcommittee. IEEE reliability test system. *IEEE Transactions on power apparatus and systems*, (6):2047–2054, 1979.

[113] S. Rai and K.K. Aggarwal. An efficient method for reliability evaluation of a general network. *Reliability, IEEE Transactions on*, R-27(3):206–211, Aug 1978.

[114] Jose E. Ramirez-Marquez and David W. Coit. Optimization of system reliability in the presence of common cause failures. *Reliability Engineering & System Safety*, 92(10):1421 – 1434, 2007.

[115] K. Durga Rao, V. Gopika, V.V.S. Sanyasi Rao, H.S. Kushwaha, A.K. Verma, and A. Srividya. Dynamic fault tree analysis using monte carlo simulation in probabilistic safety assessment. *Reliability Engineering & System Safety*, 94(4):872 – 883, 2009.

[116] Dale M Rasmuson and Dana L Kelly. Common-cause failure analysis in event assessment. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 222(4):521–532, 2008.

[117] Marvin Rausand and HÃ Arnljot. *System reliability theory: models, statistical methods, and applications*, volume 396. John Wiley & Sons, 2004.

[118] Sean Reed. An efficient algorithm for exact computation of system and survival signatures using binary decision diagrams. *Reliability Engineering & System Safety*, 165:257–267, 2017.

[119] Ehsan Reihani, Ali Sarikhani, Moez Davodi, and Mehdi Davodi. Reliability based generator maintenance scheduling using hybrid evolutionary approach. *International Journal of Electrical Power & Energy Systems*, 42(1):434 – 439, 2012.

[120] L. F. Rocha, C. L. T. Borges, and G. N. Taranto. Reliability evaluation of active distribution networks including islanding dynamics. *IEEE Transactions on Power Systems*, 32(2):1545–1552, March 2017.

[121] V. Rosato, L. Issacharoff, F. Tiriticco, S. Meloni, S. De Porcellinis, and R. Setola. Modelling interdependent infrastructures using interacting dynamical models. *International Journal of Critical Infrastructures*, 4(1/2):63+, 2008.

[122] Enno Ruijters and Marielle Stoelinga. Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools. *Computer Science Review*, 15:29 – 62, 2015.

[123] J. H. Saleh and K. Marais. Highlights from the early (and pre-) history of reliability engineering. *Reliability Engineering & System Safety*, 91(2):249–256, 2 2006.

[124] Francisco J. Samaniego. *System Signatures and their Applications in Engineering Reliability*. Springer, New York, 2007.

[125] A. Sankarakrishnan and R. Billinton. Sequential Monte Carlo simulation for composite power system reliability analysis with time varying loads. *IEEE Transactions on Power Systems*, 10(3):1540 – 1545, 1995.

[126] Dan M. Shalev and Joseph Tiran. Condition-based fault tree analysis (CBFTA): A new method for improved fault tree analysis (FTA), reliability and safety calculations. *Reliability Engineering and System Safety*, 92:1231 – 1241, 2007.

[127] Seung Ki Shin and Poong Hyun Seong. Review of various dynamic modeling methods and development of an intuitive modeling method for dynamic systems. *Nuclear Engineering and Technology*, 40(5):375–386, 2008.

[128] M.L. Shooman. *Reliability of computer systems and netwoks: fault tolerance, analysis and design*. New York, NY, USA: Wiley, Inc, 2002.

[129] Georg Steinhauser, Alexander Brandl, and Thomas E. Johnson. Comparison of the Chernobyl and Fukushima nuclear accidents: A review of the environmental impacts. *Science of The Total Environment*, 470-471:800 – 817, 2014.

[130] Masoud Taheriyoun and Saber Moradinejad. Reliability analysis of a wastewater treatment plant using fault tree analysis and Monte Carlo simulation. *Environmental Monitoring and Assessment*, 187(1), 2015.

[131] Cher Ming Tan and Nagarajan Raghavan. A framework to practical predictive maintenance modeling for multi-state systems. *Reliability Engineering & System Safety*, 93(8):1138 – 1150, 2008.

[132] Michael T. Todinov. Analysis and optimization of repairable flow networks with complex topology. *IEEE Transactions on Reliability*, 60(1):111 – 124, 2011.

[133] Matthias CM Troffaes, Gero Walter, and Dana Kelly. A robust Bayesian approach to modeling epistemic uncertainty in common-cause failure models. *Reliability Engineering & System Safety*, 125:13–21, 2014.

[134] B Tuffin, PK Choudhary, C Hirel, and KS Trivedi. Simulation versus analytic-numeric methods: a Petri-net example. In *Proc. of the 2nd VALUETOOLS Conference*, 2007.

[135] United States - Canada Power System Outage Task Force. Final report on the august 14, 2003 blackout in the United States and Canada: Causes and recommendations. Technical report, United States and Canada, April, 2004.

[136] United States Nuclear Regulatory Commission. Backgrounder on the Three Mile Island accident. `https://www.nrc.gov/reading-rm/doc-collections/fact-sheets/3mile-isle.html`, February 2013. Online; accessed 02-04-2018.

[137] United States Nuclear Regulatory Commission. Backgrounder on probabilistic risk assessment. `https://www.nrc.gov/reading-rm/doc-collections/fact-sheets/probabilistic-risk-asses.html`, February 2016. Online; accessed 31-03-2018.

[138] M. Veeraraghavan and K.S. Trivedi. A combinatorial algorithm for performance and reliability analysis using multistate models. *IEEE Transactions on Computers*, 43(2):229–234, Feb 1994.

[139] W E Vesely, F F Goldberg, N H Roberts, and D F Haasl. Fault tree handbook. Technical Report NUREG/CR-0492, U.S. Nuclear Regulatory Commission, 1981.

[140] W E Vesely, M Stamatelatos, J Dugan, J Fragola, J Minarick, and J Railsback. Fault tree handbook with aerospace applications. Technical Report Version 1.1, NASA Office of Safety and Mission Assurance, 2002.

[141] Andrija Volkanovski, Marko Cepin, and Borut Mavko. Application of the fault tree analysis for assessment of power system reliability. *Reliability Engineering and System Safety*, 94(6):1116 – 1127, 2009.

[142] Hai Canh Vu, Phuc Do, Anne Barros, and Christophe Berenguer. Maintenance grouping strategy for multi-component systems with dynamic contexts. *Reliability Engineering & System Safety*, 132:233 – 249, 2014.

[143] Jiang-Jiang Wang, Chao Fu, Kun Yang, Xu-Tao Zhang, Guo hua Shi, and John Zhai. Reliability and availability analysis of redundant BCHP (building cooling, heating and power) system. *Energy*, 61:531–540, 2013.

[144] W. Wang, J. M. Loman, R. G. Arno, P. Vassiliou, E. R. Furlong, and D. Ogden. Reliability block diagram simulation techniques applied to the IEEE Std. 493 standard network. *Industry Applications, IEEE Transactions on*, 40(3):887–895, 2004.

[145] H. S. Winokur and L. J. Goldstein. Analysis of mission-oriented systems. *IEEE Transactions on Reliability*, R-18(4):144–148, Nov 1969.

[146] Liudong Xing and Yuanshun Dai. A new decision-diagram-based method for efficient analysis on multistate systems. *Dependable and Secure Computing, IEEE Transactions on*, 6(3):161–174, July 2009.

[147] Wei-Chang Yeh. An improved sum-of-disjoint-products technique for the symbolic network reliability analysis with known minimal paths. *Reliability Engineering & System Safety*, 92(2):260 – 268, 2007.

[148] Wei-Chang Yeh. A simple minimal path method for estimating the weighted multi-commodity multistate unreliable networks reliability. *Reliability Engineering & System Safety*, 93(1):125 – 136, 2008.

[149] Wei-Chang Yeh. An improved method for multistate flow network reliability with unreliable nodes and a budget constraint based on path set. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 41(2):350–355, 2011.

[150] Wei-Chang Yeh. A fast algorithm for quickest path reliability evaluations in multi-state flow networks. *Reliability, IEEE Transactions on*, 64(4):1175–1184, Dec 2015.

[151] Wei-Chang Yeh. An improved sum-of-disjoint-products technique for symbolic multi-state flow network reliability. *Reliability, IEEE Transactions on*, 64(4):1185–1193, Dec 2015.

[152] Wei-Chang Yeh, Yi-Cheng Lin, Y.Y. Chung, and Mingchang Chih. A particle swarm optimization approach based on Monte Carlo simulation for solving the complex network reliability problem. *Reliability, IEEE Transactions on*, 59(1):212–221, March 2010.

[153] Gu Yingkui and Li Jing. Multi-state system reliability: A new and systematic review. *Procedia Engineering*, 29(0):531–536, 2012.

[154] B. Yssaad and A. Abene. Rational reliability centered maintenance optimization for power distribution systems. *International Journal of Electrical Power & Energy Systems*, 73:350 – 360, 2015.

[155] Mo Yuchang, Xing Liudong, Suprasad V. Amari, and Joanne Bechta Dugan. Efficient analysis of multi-state k-out-of-n systems. *Reliability Engineering and System Safety*, 2014.

[156] X. Zang, Dazhi Wang, Hairong Sun, and K.S. Trivedi. A BDD-based algorithm for analysis of multistate systems with multistate components. *Computers, IEEE Transactions on*, 52(12):1608–1618, Dec 2003.

[157] Z. Zhou and Q. Zhang. Model event/fault trees with dynamic uncertain causality graph for better probabilistic safety assessment. *IEEE Transactions on Reliability*, PP(99):1–11, 2017.

[158] Rae Zimmerman. Social implications of infrastructure network interactions. *Journal of Urban Technology*, 8(3):97–119, 2001.

[159] E. Zio and G. Sansavini. Modeling interdependent network systems for identifying cascade-safe operating margins. *IEEE Transactions on Reliability*, 60(1):94–101, March 2011.

[160] Enrico Zio. Challenges in the vulnerability and risk analysis of critical infrastructures. *Reliability Engineering & System Safety*, 152:137 – 150, 2016.

[161] Enrico Zio, Piero Baraldi, and Edoardo Patelli. Assessment of the availability of an offshore installation by Monte Carlo simulation. *International Journal of Pressure Vessels and Piping*, 83(4):312 – 320, 2006.

[162] Ming J. Zuo, Zhigang Tian, and Hong-Zhong Huang. An efficient method for reliability evaluation of multistate networks given all minimal path vectors. *IIE Transactions*, 39(8):811–817, 2007.